

# **QorIQ LS1043A Reference Manual**

Supports LS1023A, LS1043A\_AEC-Q100

Document Number: LS1043ARM  
Rev. 6, 07/2020





## Contents

Section number	Title	Page
	<b>Chapter 1 Overview</b>	
1.1	Introduction.....	51
1.2	Features summary.....	52
1.3	Application examples.....	53
1.3.1	Multi-service branch office router.....	53
1.3.2	Security appliance/UTM.....	54
1.4	Module features .....	55
1.4.1	Arm® Cortex®-A53 core.....	55
1.4.2	System memory management unit MMU-500.....	56
1.4.3	Arm CoreLink CCI-400 cache coherent interconnect.....	57
1.4.4	PreBoot loader (PBL) and nonvolatile memory interfaces.....	57
1.4.5	DDR memory controllers.....	58
1.4.6	Enhanced direct memory access (eDMA) and direct memory access multiplexer (DMAMUX).....	58
1.4.7	DUART.....	59
1.4.8	FlexTimer module (FTM).....	59
1.4.9	Integrated flash controller (IFC).....	59
1.4.10	Universal Serial Bus (USB) controllers and PHY.....	61
1.4.11	High speed I/O interfaces.....	61
1.4.12	QUICC Engine (QE).....	63
1.4.13	Enhanced secure digital host controller and SDIO.....	63
1.4.14	Security Engine (SEC).....	63
1.4.15	Inter-Integrated Circuit (I2C).....	64
1.4.16	Low Power Universal asynchronous receiver/ transmitter (LPUART).....	64
1.4.17	Quad Serial Peripheral Interface (QuadSPI).....	65
1.4.18	Queue Direct Memory Access Controller (qDMA).....	65
1.4.19	Serial Peripheral Interface (SPI).....	65

<b>Section number</b>	<b>Title</b>	<b>Page</b>
1.4.20	Watchdog Timer (WDOG).....	66
<b>Chapter 2 Memory Map</b>		
2.1	Memory map overview.....	67
2.2	Fixed memory map.....	67
2.2.1	DDR remapping.....	69
2.3	CCSR address map.....	70
2.4	Source ID Assignments.....	75
<b>Chapter 3 Signal Descriptions</b>		
3.1	Signals introduction.....	77
3.2	Signals overview.....	77
3.3	Configuration signals sampled at reset .....	108
3.4	Signal multiplexing details.....	109
3.4.1	UART, GPIO, FTM, and LPUART signal multiplexing.....	109
3.4.2	ASLEEP and GPIO1 signal multiplexing.....	110
3.4.3	RTC and GPIO1 signal multiplexing.....	111
3.4.4	eSDHC, GPIO2, and GPIO4 signal multiplexing.....	111
3.4.5	External IRQ, QE, and GPIO1 signal multiplexing.....	112
3.4.6	SPI, eSDHC, USB and GPIO2 signal multiplexing.....	114
3.4.7	IFC, QuadSPI, FTM and GPIO2 signal multiplexing.....	115
3.4.8	Ethernet controller 1, FTM1, and GPIO3 signal multiplexing.....	117
3.4.9	Ethernet controller 2, GPIO3, FTM2, and IEEE1588 signal multiplexing.....	118
3.4.10	Ethernet management interface1 and GPIO3 signal multiplexing.....	119
3.4.11	Ethernet management interface2 and GPIO4 signal multiplexing.....	119
3.4.12	I2C2, GPIO4, FTM, QE and eSDHC signal multiplexing.....	120
3.4.13	USB and GPIO4 signal multiplexing.....	120
3.5	Output Signal States During Reset.....	121

## Chapter 4 Reset, Clocking, and Initialization

<b>Section number</b>	<b>Title</b>	<b>Page</b>
4.1	Reset, clocking, and initialization overview.....	123
4.2	External signal descriptions.....	123
4.2.1	System control signals.....	124
4.2.2	External clock signals.....	125
4.3	Clocking Memory Map.....	126
4.3.1	Core cluster n clock control/status register (Clocking_CLKCCSR).....	127
4.3.2	Clock generator n hardware accelerator control/status register (Clocking_CLnKCGHWACSR).....	128
4.3.3	PLL cluster n general status register (Clocking_PLLCnGSR).....	130
4.3.4	Platform clock domain control/status register (Clocking_CLKPCSR).....	132
4.3.5	Platform PLL general status register (Clocking_PLLPGSR).....	133
4.3.6	DDR PLL general status register (Clocking_PLLDGSR).....	135
4.4	Functional description.....	136
4.4.1	Power-on reset sequence.....	136
4.4.2	Hard reset sequence.....	139
4.4.3	Core soft reset.....	140
4.4.4	RCW state timing.....	140
4.4.5	Power-on reset configuration.....	141
4.4.6	Reset configuration word (RCW).....	146
4.4.7	Clocking.....	165

## **Chapter 5 Interrupt Assignments**

5.1	Introduction.....	173
5.2	Internal interrupt sources.....	173

## **Chapter 6 Arm Modules**

6.1	Introduction.....	181
6.2	Arm® Cortex®-A53 core.....	182
6.3	Arm generic interrupt controller (GIC-400).....	183
6.4	System memory management unit (MMU-500).....	184

<b>Section number</b>	<b>Title</b>	<b>Page</b>
<b>Chapter 7 CSU, OCRAM, and MSCM</b>		
7.1	Central Security Unit.....	187
7.1.1	CSU Memory Map/Register Definition.....	187
7.1.2	Initialization Policy.....	198
7.2	On-Chip RAM memory controller (OCRAM).....	198
7.3	Miscellaneous System Control Module (MSCM) .....	198
7.3.1	MSCM Access Control and TrustZone Security (ACTZS)Memory Map/Register Definition.....	199
7.3.2	ACTZS CSLn Fail Status Capture Registers (Memory Map/Register Definition).....	207
<b>Chapter 8 System Counter</b>		
8.1	System counter.....	223
8.1.1	Secure system counter memory map/register definition.....	223
8.1.2	Non-Secure system counter memory map/register definition.....	226
<b>Chapter 9 Interconnect Fabric</b>		
9.1	Introduction.....	229
9.2	QoS generators.....	229
9.2.1	Fixed QoS generator.....	229
9.2.2	Limiter QoS generator.....	229
9.2.3	Regulator QoS generator.....	230
9.3	IF Memory Map/Register Definition.....	230
9.3.1	Priority qdma (IF_prio_qdma_read_only).....	231
9.3.2	Mode qdma (IF_mod_qdma_read_only).....	232
9.3.3	Bandwidth qdma (IF_bw_qdma_read_only).....	232
9.3.4	Saturation qdma (IF_sat_qdma_read_only).....	233
9.3.5	ExtControl qdma (IF_ext_cntrl_qdma_read_only).....	233
9.3.6	Priority qdma (IF_prio_qdma_write_only).....	234
9.3.7	Mode qdma (IF_mod_qdma_write_only).....	234
9.3.8	Bandwidth qdma (IF_bw_qdma_write_only).....	235

<b>Section number</b>	<b>Title</b>	<b>Page</b>
9.3.9	Saturation_qdma (IF_sat_qdma_write_only).....	235
9.3.10	ExtControl_qdma (IF_ext_cntrl_qdma_write_only).....	236
9.3.11	Priority_pex (IF_prio_pex_read_only).....	236
9.3.12	Mode_pex (IF_mod_pex_read_only).....	237
9.3.13	Bandwidth_pex (IF_bw_pex_read_only).....	238
9.3.14	Saturation_pex (IF_sat_pex_read_only).....	238
9.3.15	ExtControl_pex_ro (IF_ext_cntrl_pex_ro).....	239
9.3.16	Priority_pex (IF_prio_pex_write_only).....	239
9.3.17	Mode_pex (IF_mod_pex_write_only).....	240
9.3.18	Bandwidth_pex (IF_bw_smmu_pex_write_only).....	241
9.3.19	Saturation_pex (IF_sat_pex_write_only).....	241
9.3.20	ExtControl_pex_wo (IF_ext_cntrl_pex_write_only).....	242

## **Chapter 10 Cache Coherent Interconnect (CCI-400)**

10.1	The CCI-400 module as implemented on the chip.....	243
10.1.1	LS1043A CCI module integration.....	243
10.1.2	Connections to the CCI-400 interconnect.....	244
10.1.3	Snoop transaction configurations.....	245
10.2	CCI400 register descriptions.....	246
10.2.1	CCI400 Registers memory map.....	246
10.2.2	Control Override Register (Control_OVERRIDE_Register).....	248
10.2.3	Speculation Control Register (Speculation_Control_Register).....	250
10.2.4	Secure Access Register (Secure_Access_Register).....	252
10.2.5	Status Register (Status_Register).....	253
10.2.6	Inprecise Error Register (Inprecise_Error_Register).....	255
10.2.7	Snoop Control Registers (Snoop_Control_Register_S0 - Snoop_Control_Register_S4).....	256
10.2.8	Shareable Override Registers (Shareable_Override_Register_S0 - Shareable_Override_Register_S4)....	259
10.2.9	Read Channel QoS Value Override Register (Read_Qos_Override_Register_S0 - Read_Qos_Override_Register_S4).....	260

<b>Section number</b>	<b>Title</b>	<b>Page</b>
10.2.10	Write Qos Override Register (Write_Qos_Override_Register_S0 - Write_Qos_Override_Register_S4)..	262
10.2.11	Qos Control Register (Qos_Control_Register_S0 - Qos_Control_Register_S4).....	264
10.2.12	Max OT Registers (Max_OT_Register_S0 - Max_OT_Register_S4).....	267
10.2.13	Regulator Target Registers (Target_Latency_Register_S0 - Target_Latency_Register_S4).....	269
10.2.14	QoS Range Register (Qos_Range_Register_S0 - Qos_Range_Register_S4).....	270
10.2.15	QoS Regulator Scale Factor Registers (Latency_Regulation_Register_S0 - Latency_Regulation_Regis ter_S4).....	272
10.3	Functional Description.....	274
10.3.1	About the functions.....	274
10.3.2	Snoop connectivity and control.....	274
10.3.3	Speculative fetch.....	275
10.3.4	Security.....	276
10.3.5	Error responses.....	277
10.3.6	Barriers.....	278
10.3.7	DVM messages.....	278
10.3.8	Quality of Service.....	279

## Chapter 11 Arm CoreLink™ TrustZone Address Space Controller TZC-380

11.1	Introduction.....	285
11.1.1	Overview.....	285
11.2	Miscellaneous signal descriptions.....	286
11.3	Register Descriptions.....	287
11.3.1	TZASC memory map.....	288
11.3.2	Configuration Register (configuration).....	289
11.3.3	Action register (action).....	291
11.3.4	Lockdown Range Register (lockdown_range).....	292
11.3.5	Lockdown Select Register (lockdown_select).....	294
11.3.6	Interrupt Status Register (int_status).....	296
11.3.7	Interrupt Clear Register (int_clear).....	297

<b>Section number</b>	<b>Title</b>	<b>Page</b>
11.3.8	Fail Address Low Register (fail_address_low).....	298
11.3.9	Fail Address High Register (fail_address_high).....	299
11.3.10	Fail Control Register (fail_control).....	300
11.3.11	Fail ID Register (fail_id).....	301
11.3.12	Speculation Control Register (speculation_control).....	302
11.3.13	Security Inversion Register (security_inversion_en).....	304
11.3.14	Region Setup Low 0 Register (region_setup_low_0).....	305
11.3.15	Region Setup High 0 Register (region_setup_high_0).....	306
11.3.16	Region Attributes 0 Register (region_attributes_0).....	307
11.3.17	Region Setup Low 1 Register (region_setup_low_1).....	308
11.3.18	Region Setup High 1 Register (region_setup_high_1).....	309
11.3.19	Region Attributes 1 Register (region_attributes_1).....	310
11.3.20	Region Setup Low 2 Register (region_setup_low_2).....	314
11.3.21	Region Setup High 2 Register (region_setup_high_2).....	315
11.3.22	Region Attributes 2 Register (region_attributes_2).....	316
11.3.23	Region Setup Low 3 Register (region_setup_low_3).....	319
11.3.24	Region Setup High 3 Register (region_setup_high_3).....	320
11.3.25	Region Attributes 3 Register (region_attributes_3).....	321
11.3.26	Region Setup Low 4 Register (region_setup_low_4).....	325
11.3.27	Region Setup High 4 Register (region_setup_high_4).....	326
11.3.28	Region Attributes 4 Register (region_attributes_4).....	327
11.3.29	Region Setup Low 5 Register (region_setup_low_5).....	330
11.3.30	Region Setup High 5 Register (region_setup_high_5).....	331
11.3.31	Region Attributes 5 Register (region_attributes_5).....	332
11.3.32	Region Setup Low 6 Register (region_setup_low_6).....	336
11.3.33	Region Setup High 6 Register (region_setup_high_6).....	337
11.3.34	Region Attributes 6 Register (region_attributes_6).....	338
11.3.35	Region Setup Low 7 Register (region_setup_low_7).....	341
11.3.36	Region Setup High 7 Register (region_setup_high_7).....	342

<b>Section number</b>	<b>Title</b>	<b>Page</b>
11.3.37	Region Attributes 7 Register (region_attributes_7).....	343
11.3.38	Region Setup Low 8 Register (region_setup_low_8).....	347
11.3.39	Region Setup High 8 Register (region_setup_high_8).....	348
11.3.40	Region Attributes 8 Register (region_attributes_8).....	349
11.3.41	Region Setup Low 9 Register (region_setup_low_9).....	352
11.3.42	Region Setup High 9 Register (region_setup_high_9).....	353
11.3.43	Region Attributes 9 Register (region_attributes_9).....	354
11.3.44	Region Setup Low 10 Register (region_setup_low_10).....	358
11.3.45	Region Setup High 10 Register (region_setup_high_10).....	359
11.3.46	Region Attributes 10 Register (region_attributes_10).....	360
11.3.47	Region Setup Low 11 Register (region_setup_low_11).....	363
11.3.48	Region Setup High 11 Register (region_setup_high_11).....	364
11.3.49	Region Attributes 11 Register (region_attributes_11).....	365
11.3.50	Region Setup Low 12 Register (region_setup_low_12).....	369
11.3.51	Region Setup High 12 Register (region_setup_high_12).....	370
11.3.52	Region Attributes 12 Register (region_attributes_12).....	371
11.3.53	Region Setup Low 13 Register (region_setup_low_13).....	374
11.3.54	Region Setup High 13 Register (region_setup_high_13).....	375
11.3.55	Region Attributes 13 Register (region_attributes_13).....	376
11.3.56	Region Setup Low 14 Register (region_setup_low_14).....	380
11.3.57	Region Setup High 14 Register (region_setup_high_14).....	381
11.3.58	Region Attributes 14 Register (region_attributes_14).....	382
11.3.59	Region Setup Low 15 Register (region_setup_low_15).....	385
11.3.60	Region Setup High 15 Register (region_setup_high_15).....	386
11.3.61	Region Attributes 15 Register (region_attributes_15).....	387
11.3.62	Integration Test Control Register (itcrg).....	391
11.3.63	Integration Test Input Register (itip).....	392
11.3.64	Integration Test Output Register (itop).....	393
11.4	Functional description.....	394

Section number	Title	Page
11.4.1	Functional operation.....	395
11.4.2	Constraints of use.....	406

## Chapter 12 Supplemental Configuration Unit

12.1	Introduction .....	409
12.2	Overview .....	409
12.3	SCFG Memory Map/Register Definition.....	409
12.3.1	USB1 Parameter 1 Control Register (SCFG_USB1PRM1CR).....	412
12.3.2	USB1 Parameter 2 Control Register (SCFG_USB1PRM2CR).....	414
12.3.3	USB1 Parameter 3 Control Register (SCFG_USB1PRM3CR).....	415
12.3.4	USB2 Parameter 1 Control Register (SCFG_USB2PRM1CR).....	416
12.3.5	USB2 Parameter 2 Control Register (SCFG_USB2PRM2CR).....	418
12.3.6	USB2 Parameter3 Control Register (SCFG_USB2PRM3CR).....	419
12.3.7	USB3 Parameter 1 Control Register (SCFG_USB3PRM1CR).....	420
12.3.8	USB3 Parameter 2 Control Register (SCFG_USB3PRM2CR).....	422
12.3.9	USB3 Parameter 3 Control Register (SCFG_USB3PRM3CR).....	423
12.3.10	USB2 ICID Register (SCFG_USB2_ICID).....	424
12.3.11	USB3 ICID Register (SCFG_USB3_ICID).....	425
12.3.12	qDMA ICID Register (SCFG_DMA_ICID).....	426
12.3.13	SATA ICID Register (SCFG_SATA_ICID).....	427
12.3.14	USB1 ICID Register (SCFG_USB1_ICID).....	428
12.3.15	QE ICID Register (SCFG_QE_ICID).....	429
12.3.16	eSDHC ICID Register (SCFG_SDHC_ICID).....	430
12.3.17	eDMA ICID Register (SCFG_eDMA_ICID).....	431
12.3.18	ETR ICID Register (SCFG_ETR_ICID).....	432
12.3.19	Core 0 soft reset Register (SCFG_CORE0_SFT_RST).....	433
12.3.20	Core 1 soft reset Register (SCFG_CORE1_SFT_RST).....	434
12.3.21	Core 2 soft reset Register (SCFG_CORE2_SFT_RST).....	435
12.3.22	Core 3soft reset Register (SCFG_CORE3_SFT_RST).....	436

<b>Section number</b>	<b>Title</b>	<b>Page</b>
12.3.23	PEX PME control register (SCFG_PEXPMECR).....	437
12.3.24	FTM chain configuration (SCFG_FTM_CHAIN_CONFIG).....	438
12.3.25	ALTCBAR Register (SCFG_ALTCBAR).....	439
12.3.26	QSPI CONFIG Register (SCFG_QSPI_CFG).....	439
12.3.27	QOS1 Register (SCFG_QOS1).....	440
12.3.28	QOS2 Register (SCFG_QOS2).....	441
12.3.29	GIC-400 Address 64K Page Alignment Register (SCFG_GIC400_ADDR_ALIGN_64K).....	442
12.3.30	Debug ICID Register (SCFG_Debug_ICID).....	443
12.3.31	Snoop Configuration Register (SCFG_SNPCNFGCR).....	444
12.3.32	Interrupt Polarity Register (SCFG_INTPCR).....	446
12.3.33	CORE Soft Reset Enable Register (SCFG_CORESRENCR).....	448
12.3.34	Core 0 Reset Vector Base Address0 (SCFG_RVBAR0_0).....	448
12.3.35	Core 0 Reset Vector Base Address1 (SCFG_RVBAR0_1).....	449
12.3.36	Core 1 Reset Vector Base Address0 (SCFG_RVBAR1_0).....	450
12.3.37	Core 1 Reset Vector Base Address1 (SCFG_RVBAR1_1).....	450
12.3.38	Core 2 Reset Vector Base Address0 (SCFG_RVBAR2_0).....	451
12.3.39	Core 2 Reset Vector Base Address1 (SCFG_RVBAR2_1).....	451
12.3.40	Core 3 Reset Vector Base Address0 (SCFG_RVBAR3_0).....	452
12.3.41	Core 3 Reset Vector Base Address1 (SCFG_RVBAR3_1).....	452
12.3.42	Core Low Power Mode Control Status Register (SCFG_LPMCSR).....	453
12.3.43	ECGTX Clock Mux Control Register (SCFG_ECGTXCMCR).....	455
12.3.44	SDHC IO VSEL Control Register (SCFG_SDHCIOVSELCR).....	456
12.3.45	Extended RCW PinMux Control Register (SCFG_RCWPMUXCR0).....	457
12.3.46	USB DRVVBUS Control Register (SCFG_USBDRVVBUS_SELCR).....	459
12.3.47	USB PWRFAULTControl Register (SCFG_USBPWRFAULT_SELCR).....	459
12.3.48	USB PHY1 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR1).....	460
12.3.49	USB PHY2 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR2).....	461
12.3.50	USB PHY3 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR3).....	462
12.3.51	Retention Request Control Register (SCFG_RETREQCR).....	463

<b>Section number</b>	<b>Title</b>	<b>Page</b>
12.3.52	CORE PM Control Register (SCFG_COREPMCR).....	464
12.3.53	SCRATCHRWn - Scratch Read Write Registers (SCFG_SCRATCHRWn).....	464
12.3.54	Core Boot Control Register (SCFG_COREBCR).....	465
12.3.55	Shared Message Signaled Interrupt Index Register (SCFG_G0MSIIR).....	466
12.3.56	Shared Message Signaled Interrupt Register (SCFG_G0MSIR1).....	468
12.3.57	Shared Message Signaled Interrupt Register (SCFG_G0MSIR2).....	468
12.3.58	Shared Message Signaled Interrupt Register (SCFG_G0MSIR3).....	469
12.3.59	Shared Message Signaled Interrupt Register (SCFG_G0MSIR4).....	469
12.3.60	Shared Message Signaled Interrupt Index Register (SCFG_G1MSIIR).....	470
12.3.61	Shared Message Signaled Interrupt Register (SCFG_G1MSIR1).....	472
12.3.62	Shared Message Signaled Interrupt Register (SCFG_G1MSIR2).....	472
12.3.63	Shared Message Signaled Interrupt Register (SCFG_G1MSIR3).....	473
12.3.64	Shared Message Signaled Interrupt Register (SCFG_G1MSIR4).....	473
12.3.65	Shared Message Signaled Interrupt Index Register (SCFG_G2MSIIR).....	474
12.3.66	Shared Message Signaled Interrupt Register (SCFG_G2MSIR1).....	476
12.3.67	Shared Message Signaled Interrupt Register (SCFG_G2MSIR2).....	476
12.3.68	Shared Message Signaled Interrupt Register (SCFG_G2MSIR3).....	477
12.3.69	Shared Message Signaled Interrupt Register (SCFG_G2MSIR4).....	477

## Chapter 13 Device Configuration and Pin Control

13.1	Device Configuration and Pin Control Introduction.....	479
13.2	Features.....	479
13.3	Device Configuration/Pin Control Memory Map.....	480
13.3.1	POR Status Register 1 (DCFG_CCSR_PORSR1).....	485
13.3.2	POR Status Register 2 (DCFG_CCSR_PORSR2).....	486
13.3.3	General-Purpose POR Configuration Register (DCFG_CCSR_GPPORCR1).....	488
13.3.4	Device Disable Register 1 (DCFG_CCSR_DEVDISR1).....	488
13.3.5	Device Disable Register 2 (DCFG_CCSR_DEVDISR2).....	490
13.3.6	Device Disable Register 3 (DCFG_CCSR_DEVDISR3).....	492

<b>Section number</b>	<b>Title</b>	<b>Page</b>
13.3.7	Device Disable Register 4 (DCFG_CCSR_DEVDISR4).....	493
13.3.8	Device Disable Register 5 (DCFG_CCSR_DEVDISR5).....	494
13.3.9	Core Disable Register (DCFG_CCSR_COREDISR).....	497
13.3.10	System Version Register (DCFG_CCSR_SVR).....	499
13.3.11	Reset Control Register (DCFG_CCSR_RSTCR).....	500
13.3.12	Reset Request Preboot Loader Status Register (DCFG_CCSR_RSTRQPBLSR).....	501
13.3.13	Reset Request Mask Register (DCFG_CCSR_RSTRQMR1).....	502
13.3.14	Reset Request Status Register (DCFG_CCSR_RSTRQSR1).....	504
13.3.15	Boot Release Register (DCFG_CCSR_BRR).....	508
13.3.16	Reset Control Word Status Register n (DCFG_CCSR_RCWSR $n$ ).....	509
13.3.17	Scratch Read / Write Register n (DCFG_CCSR_SCRATCHRW $n$ ).....	510
13.3.18	Scratch Read Register n (DCFG_CCSR_SCRATCHW1R $n$ ).....	511
13.3.19	Core Reset Status Register n (DCFG_CCSR_CRSTS $Rn$ ).....	511
13.3.20	DMA Control Register (DCFG_CCSR_DMACR1).....	514
13.3.21	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP $n$ ).....	515
13.3.22	Core Cluster n Topology Register (DCFG_CCSR_TP_CLUSTER $n$ ).....	516
13.3.23	DDR Clock Disable Register (DCFG_CCSR_DDRCLKDR).....	517
13.3.24	IFC Clock Disable Register (DCFG_CCSR_IFCCLKDR).....	518
13.3.25	eSDHC Polarity Configuration Register (DCFG_CCSR_SDHCP $C$ R).....	519

## Chapter 14 Run Control and Power Management (RCPM)

14.1	Introduction.....	521
14.1.1	Overview.....	521
14.1.2	The RCPM module as implemented on the chip.....	521
14.1.3	Power Management Features.....	522
14.1.4	Power Management States.....	523
14.1.5	Modes Entry and Exit for Power Management.....	524
14.1.6	Power Management States.....	531
14.1.7	Reset modes.....	533

<b>Section number</b>	<b>Title</b>	<b>Page</b>
14.2	External Signal Description.....	534
14.3	RCPM Memory Map/Register Definition.....	534
14.3.1	Thread Wait status Register (RCPM_TWITSR).....	535
14.3.2	Physical Core PH20 Status Register (RCPM_PCPH20SR).....	536
14.3.3	Physical Core PH20 Set Control Register (RCPM_PCPH20SETR).....	536
14.3.4	Physical Core PH20 Clear Control Register (RCPM_PCPH20CLRR).....	537
14.3.5	Physical Core PH20 Previous Status Register (RCPM_PCPH20PSR).....	539
14.3.6	Power Management Control and Status Register (RCPM_POWMGTCCSR).....	539
14.3.7	IP Powerdown Exception Control Register (RCPM_IPPDEXPCR).....	541
14.3.8	nIRQOUT interrupt mask register (RCPM_nIRQOUTR).....	543
14.3.9	nFIQOUT Interrupt Register (RCPM_nFIQOUTR).....	543
14.4	RCPM functional description.....	544

## **Chapter 15** **QUICC Engine Block**

15.1	Introduction.....	545
15.2	QUICC Engine block.....	545
15.3	QUICC Engine implementation details.....	546
15.3.1	System interface.....	548
15.3.2	Configuration.....	551
15.3.3	Multiplexing and timers.....	552
15.3.4	Unified communications controllers (UCCs).....	556
15.3.5	HDLC controller.....	556
15.3.6	Transparent controller.....	557
15.3.7	UCC for fast protocols.....	557
15.3.8	UCC for slow protocols.....	557
15.3.9	BISYNC mode.....	557
15.3.10	Serial interface with time-slot assigner.....	558
15.3.11	Multi-channel controller on UCC (UMCC).....	558

## **Chapter 16**

Section number	Title	Page
<b>Data Path Acceleration Architecture (DPAA) Overview and SoC DPAA Implementation</b>		
16.1	DPAA Introduction and Terms.....	559
16.2	Data Formats Used in the DPAA.....	561
16.2.1	Frame Descriptor (FD).....	562
16.2.2	Multi-Buffer Frames.....	563
16.2.3	Single-Buffer Frames.....	566
16.2.4	Compound Frames.....	566
16.2.5	Simple Frames.....	568
16.2.6	Frame Format Codes.....	568
16.2.7	Frame Formats Supported by Accelerators.....	569
16.2.8	Special Values and Exceptions .....	570
16.2.9	Releasing Buffers to the BMan.....	570
16.3	Accessing Memory Using Isolation Context Identifier(ICID).....	570
16.4	Packet Walk-Through Example.....	571
16.5	LS1043A-Specific DPAA Implementation Details.....	573
16.5.1	Queue Manager (QMan) Implementation.....	573
16.5.2	Buffer Manager (BMan) Implementation.....	574
16.5.3	Frame Manager (FMan) Implementation.....	574
16.5.4	Security and Encryption Engine (SEC) Implementation.....	575

## **Chapter 17 Secure Boot and Trust Architecture**

17.1	Trust architecture objectives.....	577
17.2	Characteristics and claims.....	577
17.3	Non-claims.....	579
17.4	Related resources .....	581

## **Chapter 18 DDR Memory Controller**

18.1	DDR Introduction.....	583
18.2	DDR Features.....	584
18.2.1	DDR Modes of Operation.....	585

<b>Section number</b>	<b>Title</b>	<b>Page</b>
18.3	DDR External Signal Descriptions.....	585
18.3.1	DDR Signals Overview.....	586
18.3.2	DDR Detailed Signal Descriptions .....	588
18.4	DDR register descriptions.....	593
18.4.1	DDR memory map.....	593
18.4.2	Chip select a memory bounds (CS0_BNDS - CS3_BNDS).....	596
18.4.3	Chip select a configuration (CS0_CONFIG - CS3_CONFIG).....	597
18.4.4	DDR SDRAM timing configuration 3 (TIMING_CFG_3).....	600
18.4.5	DDR SDRAM timing configuration 0 (TIMING_CFG_0).....	603
18.4.6	DDR SDRAM timing configuration 1 (TIMING_CFG_1).....	607
18.4.7	DDR SDRAM timing configuration 2 (TIMING_CFG_2).....	610
18.4.8	DDR SDRAM control configuration (DDR_SDRAM_CFG).....	613
18.4.9	DDR SDRAM control configuration 2 (DDR_SDRAM_CFG_2).....	617
18.4.10	DDR SDRAM mode configuration (DDR_SDRAM_MODE).....	620
18.4.11	DDR SDRAM mode configuration 2 (DDR_SDRAM_MODE_2).....	621
18.4.12	DDR SDRAM mode control (DDR_SDRAM_MD_CNTL).....	623
18.4.13	DDR SDRAM interval configuration (DDR_SDRAM_INTERVAL).....	626
18.4.14	DDR SDRAM data initialization (DDR_DATA_INIT).....	627
18.4.15	DDR SDRAM clock control (DDR_SDRAM_CLK_CNTL).....	628
18.4.16	DDR training initialization address (DDR_INIT_ADDR).....	629
18.4.17	DDR training initialization extended address (DDR_INIT_EXT_ADDRESS).....	631
18.4.18	DDR SDRAM timing configuration 4 (TIMING_CFG_4).....	632
18.4.19	DDR SDRAM timing configuration 5 (TIMING_CFG_5).....	636
18.4.20	DDR SDRAM timing configuration 6 (TIMING_CFG_6).....	638
18.4.21	DDR SDRAM timing configuration 7 (TIMING_CFG_7).....	641
18.4.22	DDR ZQ calibration control (DDR_ZQ_CNTL).....	643
18.4.23	DDR write leveling control (DDR_WRLVL_CNTL).....	646
18.4.24	DDR Self Refresh Counter (DDR_SR_CNTR).....	650
18.4.25	DDR Register Control Words 1 (DDR_SDRAM_RCW_1).....	651

<b>Section number</b>	<b>Title</b>	<b>Page</b>
18.4.26	DDR Register Control Words 2 (DDR_SDRAM_RCW_2).....	652
18.4.27	DDR write leveling control 2 (DDR_WRLVL_CNTL_2).....	654
18.4.28	DDR write leveling control 3 (DDR_WRLVL_CNTL_3).....	658
18.4.29	DDR Register Control Words 3 (DDR_SDRAM_RCW_3).....	661
18.4.30	DDR Register Control Words 4 (DDR_SDRAM_RCW_4).....	662
18.4.31	DDR Register Control Words 5 (DDR_SDRAM_RCW_5).....	664
18.4.32	DDR Register Control Words 6 (DDR_SDRAM_RCW_6).....	665
18.4.33	DDR SDRAM mode configuration 3 (DDR_SDRAM_MODE_3).....	666
18.4.34	DDR SDRAM mode configuration 4 (DDR_SDRAM_MODE_4).....	667
18.4.35	DDR SDRAM mode configuration 5 (DDR_SDRAM_MODE_5).....	669
18.4.36	DDR SDRAM mode configuration 6 (DDR_SDRAM_MODE_6).....	670
18.4.37	DDR SDRAM mode configuration 7 (DDR_SDRAM_MODE_7).....	671
18.4.38	DDR SDRAM mode configuration 8 (DDR_SDRAM_MODE_8).....	673
18.4.39	DDR SDRAM mode configuration 9 (DDR_SDRAM_MODE_9).....	674
18.4.40	DDR SDRAM mode configuration 10 (DDR_SDRAM_MODE_10).....	675
18.4.41	DDR SDRAM mode configuration 11 (DDR_SDRAM_MODE_11).....	676
18.4.42	DDR SDRAM mode configuration 12 (DDR_SDRAM_MODE_12).....	677
18.4.43	DDR SDRAM mode configuration 13 (DDR_SDRAM_MODE_13).....	679
18.4.44	DDR SDRAM mode configuration 14 (DDR_SDRAM_MODE_14).....	680
18.4.45	DDR SDRAM mode configuration 15 (DDR_SDRAM_MODE_15).....	681
18.4.46	DDR SDRAM mode configuration 16 (DDR_SDRAM_MODE_16).....	682
18.4.47	DDR SDRAM timing configuration 8 (TIMING_CFG_8).....	684
18.4.48	DDR SDRAM control configuration 3 (DDR_SDRAM_CFG_3).....	688
18.4.49	DQ mapping register 0 (DDR_DQ_MAP0).....	691
18.4.50	DQ mapping register 1 (DDR_DQ_MAP1).....	694
18.4.51	DQ mapping register 2 (DDR_DQ_MAP2).....	695
18.4.52	DQ mapping register 3 (DDR_DQ_MAP3).....	697
18.4.53	DDR Debug Status Register 1 (DDRDSR_1).....	698
18.4.54	DDR Debug Status Register 2 (DDRDSR_2).....	699

<b>Section number</b>	<b>Title</b>	<b>Page</b>
18.4.55	DDR Control Driver Register 1 (DDRCDR_1).....	701
18.4.56	DDR Control Driver Register 2 (DDRCDR_2).....	703
18.4.57	DDR IP block revision 1 (DDR_IP_REV1).....	705
18.4.58	DDR IP block revision 2 (DDR_IP_REV2).....	706
18.4.59	DDR Memory Test Control Register (DDR_MTCR).....	707
18.4.60	DDR Memory Test Pattern n Register (DDR_MTP0 - DDR_MTP9).....	710
18.4.61	DDR Memory Test Start Extended Address (DDR_MT_ST_EXT_ADDR).....	711
18.4.62	DDR Memory Test Start Address (DDR_MT_ST_ADDR).....	712
18.4.63	DDR Memory Test End Extended Address (DDR_MT_END_EXT_ADDR).....	713
18.4.64	DDR Memory Test End Address (DDR_MT_END_ADDR).....	714
18.4.65	Memory data path error injection mask high (DATA_ERR_INJECT_HI).....	715
18.4.66	Memory data path error injection mask low (DATA_ERR_INJECT_LO).....	716
18.4.67	Memory data path error injection mask ECC (ECC_ERR_INJECT).....	717
18.4.68	Memory data path read capture high (CAPTURE_DATA_HI).....	719
18.4.69	Memory data path read capture low (CAPTURE_DATA_LO).....	719
18.4.70	Memory data path read capture ECC (CAPTURE_ECC).....	720
18.4.71	Memory error detect (ERR_DETECT).....	722
18.4.72	Memory error disable (ERR_DISABLE).....	724
18.4.73	Memory error interrupt enable (ERR_INT_EN).....	725
18.4.74	Memory error attributes capture (CAPTURE_ATTRIBUTES).....	727
18.4.75	Memory error address capture (CAPTURE_ADDRESS).....	729
18.4.76	Memory error extended address capture (CAPTURE_EXT_ADDRESS).....	730
18.4.77	Single-Bit ECC memory error management (ERR_SBE).....	731
18.5	DDR Functional Description.....	732
18.5.1	DDR SDRAM Interface Operation.....	736
18.5.2	DDR SDRAM Address Multiplexing.....	737
18.5.3	DDR SDRAM Write Timing Adjustments.....	741
18.5.4	DDR SDRAM Refresh.....	742
18.5.5	DDR Data Beat Ordering.....	744

<b>Section number</b>	<b>Title</b>	<b>Page</b>
18.5.6	Page Mode and Logical Bank Retention.....	744
18.5.7	Error Checking and Correcting (ECC).....	745
18.5.8	Error Management.....	747
18.5.9	DDR Rapid Clear of Memory.....	748
18.6	Initialization/Application Information.....	749
18.6.1	Programming Summary .....	753
18.6.2	DDR SDRAM Initialization Sequence.....	759
18.6.3	Using Forced Self-Refresh Mode to Implement a Battery-Backed RAM System.....	759

## **Chapter 19 Direct Memory Access Multiplexer (DMAMUX)**

19.1	The DMAMUX module as implemented on the chip.....	761
19.1.1	LS1043A DMAMUX module integration.....	761
19.1.2	LS1043A DMAMUX module special consideration.....	761
19.2	Introduction.....	765
19.2.1	Overview.....	765
19.2.2	Features.....	766
19.2.3	Modes of operation.....	766
19.3	External signal description.....	767
19.4	Memory map/register definition.....	767
19.4.1	Channel Configuration register (DMAMUX $x$ _CHCFG $n$ ).....	768
19.5	Functional description.....	769
19.5.1	Always-enabled DMA sources.....	769
19.6	Initialization/application information.....	770
19.6.1	Reset.....	770
19.6.2	Enabling and configuring sources.....	771

## **Chapter 20 DUART**

20.1	The DUART module as implemented on the chip.....	773
20.2	Overview.....	773

<b>Section number</b>	<b>Title</b>	<b>Page</b>
20.2.1	Features.....	774
20.2.2	Modes of operation.....	775
20.3	DUART external signal descriptions.....	775
20.4	DUART register descriptions.....	776
20.4.1	DUART memory map.....	777
20.4.2	UART divisor least significant byte register (UDLB1 - UDLB2).....	778
20.4.3	UART receiver buffer register (URBR1 - URBR2).....	779
20.4.4	UART transmitter holding register (UTHR1 - UTHR2).....	780
20.4.5	UART divisor most significant byte register (UDMB1 - UDMB2).....	781
20.4.6	UART interrupt enable register (UIER1 - UIER2).....	783
20.4.7	UART alternate function register (UAFR1 - UAFR2).....	784
20.4.8	UART FIFO control register (UFCR1 - UFCR2).....	785
20.4.9	UART interrupt ID register (UIIR1 - UIIR2).....	786
20.4.10	UART line control register (ULCR1 - ULCR2).....	789
20.4.11	UART modem control register (UMCR1 - UMCR2).....	791
20.4.12	UART line status register (ULSR1 - ULSR2).....	792
20.4.13	UART modem status register (UMSR1 - UMSR2).....	794
20.4.14	UART scratch register (USCR1 - USCR2).....	795
20.4.15	UART DMA status register (UDSR1 - UDSR2).....	796
20.5	Functional description.....	798
20.5.1	Serial interface.....	798
20.5.2	Baud-rate generator logic.....	800
20.5.3	Local loopback mode.....	801
20.5.4	Errors.....	801
20.5.5	FIFO mode.....	802
20.6	Initialization/Application information.....	803

## Chapter 21 Enhanced Direct Memory Access (eDMA)

21.1	Overview .....	805
------	----------------	-----

<b>Section number</b>	<b>Title</b>	<b>Page</b>
21.2	Introduction.....	805
21.2.1	eDMA system block diagram.....	805
21.2.2	Block parts.....	806
21.2.3	Features.....	807
21.3	Modes of operation.....	808
21.4	Memory map/register definition.....	809
21.4.1	TCD memory.....	809
21.4.2	TCD initialization.....	809
21.4.3	TCD structure.....	809
21.4.4	Reserved memory and bit fields.....	810
21.4.5	Endianness.....	810
21.4.6	Control Register (DMA_CR).....	837
21.4.7	Error Status Register (DMA_ES).....	840
21.4.8	Enable Request Register (DMA_ERQ).....	842
21.4.9	Enable Error Interrupt Register (DMA_EEI).....	846
21.4.10	Clear Enable Error Interrupt Register (DMA_CEEI).....	849
21.4.11	Set Enable Error Interrupt Register (DMA_SEEI).....	850
21.4.12	Clear Enable Request Register (DMA_CERQ).....	851
21.4.13	Set Enable Request Register (DMA_SERQ).....	852
21.4.14	Clear DONE Status Bit Register (DMA_CDNE).....	853
21.4.15	Set START Bit Register (DMA_SSRT).....	854
21.4.16	Clear Error Register (DMA_CERR).....	855
21.4.17	Clear Interrupt Request Register (DMA_CINT).....	856
21.4.18	Interrupt Request Register (DMA_INT).....	857
21.4.19	Error Register (DMA_ERR).....	860
21.4.20	Hardware Request Status Register (DMA_HRS).....	864
21.4.21	Channel n Priority Register (DMA_DCHPRI <sub>n</sub> ).....	870
21.4.22	TCD Source Address (DMA_TCD <sub>n</sub> _SADDR).....	871
21.4.23	TCD Signed Source Address Offset (DMA_TCD <sub>n</sub> _SOFF).....	872

<b>Section number</b>	<b>Title</b>	<b>Page</b>
21.4.24	TCD Transfer Attributes (DMA_TCD $n$ _ATTR).....	872
21.4.25	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD $n$ _NBYTES_MLNO).....	873
21.4.26	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD $n$ _NBYTES_MLOFFNO).....	874
21.4.27	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD $n$ _NBYTES_MLOFFYES).....	875
21.4.28	TCD Last Source Address Adjustment (DMA_TCD $n$ _SLAST).....	877
21.4.29	TCD Destination Address (DMA_TCD $n$ _DADDR).....	877
21.4.30	TCD Signed Destination Address Offset (DMA_TCD $n$ _DOFF).....	878
21.4.31	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD $n$ _CITER_ELINKYES).....	878
21.4.32	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD $n$ _CITER_ELINKNO).....	879
21.4.33	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD $n$ _DLASTSGA).....	880
21.4.34	TCD Control and Status (DMA_TCD $n$ _CSR).....	881
21.4.35	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD $n$ _BITER_ELINKYES).....	883
21.4.36	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD $n$ _BITER_ELINKNO).....	884
21.5	Functional description.....	885
21.5.1	eDMA basic data flow.....	885
21.5.2	Fault reporting and handling.....	888
21.5.3	Channel preemption.....	891
21.6	Initialization/application information.....	891
21.6.1	eDMA initialization.....	891
21.6.2	Programming errors.....	893
21.6.3	Arbitration mode considerations.....	894
21.6.4	Performing DMA transfers.....	895
21.6.5	Monitoring transfer descriptor status.....	899
21.6.6	Channel Linking.....	901

<b>Section number</b>	<b>Title</b>	<b>Page</b>
21.6.7	Dynamic programming.....	902
21.6.8	Suspend/resume a DMA channel with active hardware service requests.....	906
<b>Chapter 22</b> <b>Enhanced Secured Digital Host Controller</b>		
22.1	Overview.....	909
22.1.1	eSDHC features summary.....	911
22.1.2	Modes and operations.....	912
22.2	External signals.....	913
22.2.1	External signals overview.....	913
22.2.2	eSDHC signal descriptions.....	914
22.3	eSDHC register descriptions.....	915
22.3.1	eSDHC memory map.....	915
22.3.2	SDMA system address register/Block attributes 2 (DSADDR_BLKATTR2).....	916
22.3.3	Block attributes register (BLKATTR).....	917
22.3.4	Command argument register (CMDARG).....	919
22.3.5	Transfer type register (XFERTYP).....	920
22.3.6	Command response 0 register (CMDRSP0).....	924
22.3.7	Command response 1 register (CMDRSP1).....	925
22.3.8	Command response 2 register (CMDRSP2).....	926
22.3.9	Command Response 3 register (CMDRSP3).....	927
22.3.10	Buffer data port register (DATPORT).....	929
22.3.11	Present state register (PRSSTAT).....	930
22.3.12	Protocol control register (PROCTL).....	935
22.3.13	System Control Register when ESDHCCTL[CRS=0] (SYSCTL_ESDHCCCTL_CRS_0).....	938
22.3.14	System Control Register when ESDHCCTL[CRS=1] (SYSCTL_ESDHCCCTL_CRS_1).....	942
22.3.15	Interrupt status register (IRQSTAT).....	945
22.3.16	Interrupt status enable register (IRQSTATEN).....	950
22.3.17	Interrupt signal enable register (IRQSIGEN).....	952
22.3.18	Auto CMD Error Status Register / System Control 2 Register (AUTOCERR_SYSCTL2).....	955

<b>Section number</b>	<b>Title</b>	<b>Page</b>
22.3.19	Host controller capabilities register (HOSTCAPBLT).....	958
22.3.20	Watermark level register (WML).....	960
22.3.21	Force event register (FEVT).....	962
22.3.22	ADMA error status register (ADMAES).....	964
22.3.23	ADMA system address register (ADSADDR).....	966
22.3.24	Host controller version register (HOSTVER).....	967
22.3.25	DMA error address register (DMAERRADDR).....	969
22.3.26	DMA error attribute register (DMAERRATTR).....	970
22.3.27	Host controller capabilities register 2 (HOSTCAPBLT2).....	971
22.3.28	Tuning block control register (TBCTL).....	973
22.3.29	Tuning block status register (TBSTAT).....	974
22.3.30	Tuning block pointer register (TBPTR).....	975
22.3.31	SD direction control register (SDDIRCTL).....	976
22.3.32	SD Clock Control Register (SDCLKCTL).....	978
22.3.33	eSDHC control register (ESDHCCCTL).....	979
22.4	Functional description.....	981
22.4.1	System interface and control unit (SysICU).....	982
22.4.2	SD interface and control unit (SDICU) .....	997
22.4.3	Register bank .....	1000
22.4.4	Clock and reset module.....	1001
22.4.5	SD monitor.....	1003
22.5	Initialization/application of eSDHC.....	1008
22.5.1	Command send and response receive basic operation.....	1008
22.5.2	Card identification mode.....	1009
22.5.3	Card access.....	1014
22.5.4	Switch function.....	1027
22.5.5	ADMA operation.....	1029
22.6	Interfacing Card.....	1030
22.7	Commands for MMC/SD/SDIO and .....	1031

<b>Section number</b>	<b>Title</b>	<b>Page</b>
22.8	Software restrictions.....	1037
22.8.1	Software polling procedure.....	1038
22.8.2	Suspend operation.....	1038
22.8.3	Data port access.....	1038
22.8.4	Multi-block read.....	1038
22.8.5	ADMA address.....	1039
22.8.6	Allowed operations after stop at block gap.....	1039
22.8.7	SDIO card interrupt during soft reset.....	1039
22.8.8	Soft reset for data not allowed when SD clock is disabled.....	1039
22.8.9	Data transfer with Auto CMD12 Enable.....	1039

## **Chapter 23** **FlexTimer Module (FTM)**

23.1	The FlexTimer module as implemented on the chip.....	1041
23.1.1	LS1043A FlexTimer module integration.....	1041
23.1.2	LS1043A FlexTimer signals.....	1041
23.1.3	LS1043A FlexTimer module special consideration.....	1042
23.2	Introduction.....	1043
23.2.1	FlexTimer philosophy.....	1043
23.2.2	Features.....	1044
23.2.3	Modes of operation.....	1045
23.2.4	Block diagram.....	1046
23.3	FTM signal descriptions.....	1048
23.4	Memory map and register definition.....	1048
23.4.1	Memory map.....	1048
23.4.2	Register descriptions.....	1049
23.4.3	Status And Control (FTMx_SC).....	1061
23.4.4	Counter (FTMx_CNT).....	1062
23.4.5	Modulo (FTMx_MOD).....	1063
23.4.6	Channel (n) Status And Control (FTMx_CnSC).....	1064

<b>Section number</b>	<b>Title</b>	<b>Page</b>
23.4.7	Channel (n) Value (FTMx_CnV).....	1066
23.4.8	Counter Initial Value (FTMx_CNTIN).....	1067
23.4.9	Capture And Compare Status (FTMx_STATUS).....	1067
23.4.10	Features Mode Selection (FTMx_MODE).....	1069
23.4.11	Synchronization (FTMx_SYNC).....	1071
23.4.12	Initial State For Channels Output (FTMx_OUTINIT).....	1074
23.4.13	Output Mask (FTMx_OUTMASK).....	1075
23.4.14	Function For Linked Channels (FTMx_COMBINE).....	1077
23.4.15	Deadtime Insertion Control (FTMx_DEADTIME).....	1082
23.4.16	FTM External Trigger (FTMx_EXTTRIG).....	1083
23.4.17	Channels Polarity (FTMx_POL).....	1085
23.4.18	Fault Mode Status (FTMx_FMS).....	1087
23.4.19	Input Capture Filter Control (FTMx_FILTER).....	1089
23.4.20	Fault Control (FTMx_FLTCTRL).....	1090
23.4.21	Quadrature Decoder Control And Status (FTMx_QDCTRL).....	1092
23.4.22	Configuration (FTMx_CONF).....	1094
23.4.23	FTM Fault Input Polarity (FTMx_FLTPOL).....	1095
23.4.24	Synchronization Configuration (FTMx_SYNCONF).....	1097
23.4.25	FTM Inverting Control (FTMx_INVCTRL).....	1099
23.4.26	FTM Software Output Control (FTMx_SWOCTRL).....	1100
23.4.27	FTM PWM Load (FTMx_PWMLOAD).....	1102
23.5	Functional description.....	1103
23.5.1	Clock source.....	1104
23.5.2	Prescaler.....	1105
23.5.3	Counter.....	1105
23.5.4	Input Capture mode.....	1111
23.5.5	Output Compare mode.....	1114
23.5.6	Edge-Aligned PWM (EPWM) mode.....	1115
23.5.7	Center-Aligned PWM (CPWM) mode.....	1117

<b>Section number</b>	<b>Title</b>	<b>Page</b>
23.5.8	Combine mode.....	1119
23.5.9	Complementary mode.....	1126
23.5.10	Registers updated from write buffers.....	1127
23.5.11	PWM synchronization.....	1129
23.5.12	Inverting.....	1145
23.5.13	Software output control.....	1146
23.5.14	Deadtime insertion.....	1148
23.5.15	Output mask.....	1151
23.5.16	Fault control.....	1151
23.5.17	Polarity control.....	1155
23.5.18	Initialization.....	1156
23.5.19	Features priority.....	1156
23.5.20	Channel trigger output.....	1157
23.5.21	Initialization trigger.....	1158
23.5.22	Capture Test mode.....	1160
23.5.23	DMA.....	1161
23.5.24	Dual Edge Capture mode.....	1162
23.5.25	Quadrature Decoder mode.....	1169
23.5.26	Intermediate load.....	1174
23.5.27	Global time base (GTB).....	1176
23.6	Reset overview.....	1177
23.7	FTM Interrupts.....	1179
23.7.1	Timer Overflow Interrupt.....	1179
23.7.2	Channel (n) Interrupt.....	1179
23.7.3	Fault Interrupt.....	1179
23.8	Initialization Procedure.....	1179

## Chapter 24 General Purpose I/O (GPIO)

24.1	The GPIO module as implemented on the chip.....	1183
------	---	------

<b>Section number</b>	<b>Title</b>	<b>Page</b>
24.2	GPIO overview.....	1183
24.3	GPIO features summary.....	1184
24.4	GPIO signal descriptions.....	1185
24.5	GPIO register descriptions.....	1185
24.5.1	GPIO memory map.....	1185
24.5.2	GPIO direction register (GPDIR).....	1186
24.5.3	GPIO open drain register (GPODR).....	1187
24.5.4	GPIO data register (GPDAT).....	1188
24.5.5	GPIO interrupt event register (GPIER).....	1189
24.5.6	GPIO interrupt mask register (GPIMR).....	1190
24.5.7	GPIO interrupt control register (GPICR).....	1191
<b>Chapter 25</b>		
<b>Integrated Flash Controller (IFC)</b>		
25.1	IFC overview.....	1193
25.1.1	IFC features summary.....	1194
25.1.2	IFC modes of operation.....	1197
25.2	External signal descriptions.....	1197
25.2.1	Internal connectivity of WP/RB signal.....	1201
25.3	IFC memory map/register definition.....	1204
25.3.1	IFC Revision Control register (IFC_REV).....	1215
25.3.2	Extended Chip Select Property registers (IFC_CSPR <sub>n</sub> _EXT).....	1215
25.3.3	Chip-select Property register n (IFC_CSPR <sub>n</sub> ).....	1216
25.3.4	Address Mask register (IFC_AMAS <sub>n</sub> K).....	1218
25.3.5	Chip-Select Option register - NAND Flash Mode (IFC_CSOR <sub>n</sub> _NAND).....	1220
25.3.6	Chip-Select Option register - NOR Flash Mode (IFC_CSOR <sub>n</sub> _NOR).....	1223
25.3.7	Chip-Select Option register - GPCM (IFC_CSOR <sub>n</sub> _GPCM).....	1225
25.3.8	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR <sub>n</sub> _EXT).....	1229
25.3.9	Flash Timing register 0 for Chip Select <i>n</i> - NAND flash asyncNVDDR mode (IFC_FTIM0_CS <sub>n</sub> _NAND).....	1231

<b>Section number</b>	<b>Title</b>	<b>Page</b>
25.3.10	Flash Timing register 0 for Chip Select <i>n</i> - NAND flash Asynchronous Mode (IFC_FTIM0_CS <i>n</i> _NAND_ASYNC_MODE).....	1233
25.3.11	Flash Timing register 0 for CS <i>n</i> - NOR Flash Mode (IFC_FTIM0_CS <i>n</i> _NOR).....	1235
25.3.12	Flash Timing register 0 for CS <i>n</i> - Normal GPCM Mode (IFC_FTIM0_CS <i>n</i> _GPCM).....	1236
25.3.13	Flash Timing register 1 for Chip-Select <i>n</i> - NAND Flash NVDDR Mode (IFC_FTIM1_CS <i>n</i> _NAND)....	1237
25.3.14	Flash Timing register 1 for Chip Select <i>n</i> - Asynchronous Mode (IFC_FTIM1_CS <i>n</i> _NAND_ASYNC_MODE).....	1238
25.3.15	Flash Timing register 1 for CS <i>n</i> - NOR Flash Mode (IFC_FTIM1_CS <i>n</i> _NOR).....	1239
25.3.16	Flash Timing register 1 for CS <i>n</i> - Normal GPCM Mode (IFC_FTIM1_CS <i>n</i> _GPCM).....	1240
25.3.17	Flash Timing register 2 for Chip Select <i>n</i> - NAND Flash NVDDR Mode (IFC_FTIM2_CS <i>n</i> _NAND)....	1241
25.3.18	Flash Timing register 2 for Chip Select <i>n</i> - NAND Flash Asynchronous Mode (IFC_FTIM2_CS <i>n</i> _NAND_ASYNC_MODE).....	1241
25.3.19	Flash Timing register 2 for CS <i>n</i> - NOR Flash Mode (IFC_FTIM2_CS <i>n</i> _NOR).....	1243
25.3.20	Flash Timing register 2 for CS <i>n</i> - Normal GPCM Mode (IFC_FTIM2_CS <i>n</i> _GPCM).....	1244
25.3.21	Flash Timing register 3 for Chip Select <i>n</i> - NAND Flash Mode (IFC_FTIM3_CS <i>n</i> _NAND).....	1245
25.3.22	Flash Timing register 3 for Chip Select <i>n</i> - NAND Flash Asynchronous Mode (IFC_FTIM3_CS <i>n</i> _NAND_ASYNC_MODE).....	1246
25.3.23	Flash Timing register 3 for CS <i>n</i> - NOR Flash Mode (IFC_FTIM3_CS_NOR).....	1246
25.3.24	Flash Timing register 3 for CS <i>n</i> - Normal GPCM Mode (IFC_FTIM3_CS <i>n</i> _GPCM).....	1247
25.3.25	Ready Busy Status for each Chip Select (IFC_RB_STAT).....	1248
25.3.26	General Control register (IFC_GCR).....	1249
25.3.27	Common Event and Error Status register (IFC_CM_EVTER_STAT).....	1250
25.3.28	Common Event and Error Enable register (IFC_CM_EVTER_EN).....	1251
25.3.29	Common Event and Error Interrupt Enable register (IFC_CM_EVTER_INTR_EN).....	1252
25.3.30	Common Transfer Error Attributes register 0 (IFC_CM_ERATTR0).....	1252
25.3.31	Common Transfer Error Attributes register 1 (IFC_CM_ERATTR1).....	1254
25.3.32	Clock Control register (IFC_CCR).....	1254
25.3.33	Clock Status register (IFC_CSR).....	1256
25.3.34	DDR Clock Control register (IFC_DDR_CCR).....	1257
25.3.35	NAND Configuration register (IFC_NCFGR).....	1259

<b>Section number</b>	<b>Title</b>	<b>Page</b>
25.3.36	NAND Flash Command register 0 (IFC_NAND_FCR0).....	1261
25.3.37	NAND Flash Command register 1 (IFC_NAND_FCR1).....	1262
25.3.38	Flash Row Address register n (IFC_ROW $n$ ).....	1262
25.3.39	Flash COL Address register n (IFC_COL $n$ ).....	1263
25.3.40	Flash COL Address register for 2 KB Large-Page Device (IFC_COL $n$ _2KB).....	1264
25.3.41	Flash COL Address register for 4 KB Large-Page Device (IFC_COL $n$ _4KB).....	1265
25.3.42	Flash COL Address register for 8 KB Large-Page Device (IFC_COL $n$ _8KB).....	1266
25.3.43	Flash Byte Count register for NAND Flash (IFC_NAND_BC).....	1267
25.3.44	NAND Flash Instruction register 0 (IFC_NAND_FIR0).....	1268
25.3.45	NAND Flash Instruction register 1 (IFC_NAND_FIR1).....	1270
25.3.46	NAND Flash Instruction register 2 (IFC_NAND_FIR2).....	1271
25.3.47	NAND Chip-Select register (IFC_NAND_CSEL).....	1272
25.3.48	NAND Operation Sequence Start (IFC_NANDSEQ_STRT).....	1272
25.3.49	NAND Event and Error Status register (IFC_NAND_EVTER_STAT).....	1274
25.3.50	NAND Page Read Completion Event Status register (IFC_PGRDCMPL_EVT_STAT).....	1276
25.3.51	NAND Event and Error Enable register (IFC_NAND_EVTER_EN).....	1278
25.3.52	NAND Event and Error Interrupt Enable register (IFC_NAND_EVTER_INTR_EN).....	1280
25.3.53	NAND Transfer Error Attributes register 0 (IFC_NAND_ERATTR0).....	1281
25.3.54	NAND Transfer Error Attributes register 1 (IFC_NAND_ERATTR1).....	1282
25.3.55	NAND Flash Status register (IFC_NAND_FSR).....	1283
25.3.56	ECC Status and Result of Flash Operation register 0 (IFC_ECCSTAT0).....	1283
25.3.57	ECC Status and Result of Flash Operation register 1 (IFC_ECCSTAT1).....	1285
25.3.58	ECC Status and Result of Flash Operation register 2 (IFC_ECCSTAT2).....	1286
25.3.59	ECC Status and Result of Flash Operation register 3 (IFC_ECCSTAT3).....	1287
25.3.60	NAND Control register (IFC_NANDCR).....	1288
25.3.61	NAND Autoboot Trigger register (IFC_NAND_AUTOBOOT_TRGR).....	1288
25.3.62	NAND Flash Memory Data register (IFC_NAND_MDR).....	1290
25.3.63	Nand DLL Low Config 0 Register (IFC_NAND_DLL_LOW_CFG0).....	1291
25.3.64	Nand DLL Low Config 1 Register (IFC_NAND_DLL_LOW_CFG1).....	1292

<b>Section number</b>	<b>Title</b>	<b>Page</b>
25.3.65	NAND DLL Low Status Register (IFC_NAND_DLL_LOW_STAT).....	1294
25.3.66	NOR Event and Error Status register (IFC_NOR_EVTER_STAT).....	1295
25.3.67	NOR Event and Error Enable register (IFC_NOR_EVTER_EN).....	1297
25.3.68	NOR Event and Error Interrupt enable register (IFC_NOR_EVTER_INTR_EN).....	1299
25.3.69	NOR Transfer Error Attributes register 0 (IFC_NOR_ERATTR0).....	1300
25.3.70	NOR Transfer Error Attribute register 1 (IFC_NOR_ERATTR1).....	1301
25.3.71	NOR Transfer Error Attribute register 2 (IFC_NOR_ERATTR2).....	1302
25.3.72	NOR Control register (IFC_NORCR).....	1302
25.3.73	GPCM Event and Error Status register (IFC_GPCM_EVTER_STAT).....	1304
25.3.74	GPCM Event and Error Enable register (IFC_GPCM_EVTER_EN).....	1306
25.3.75	GPCM Event and Error Interrupt enable register (IFC_GPCM_EVTER_INTR_EN).....	1307
25.3.76	GPCM Transfer Error Attributes register 0 (IFC_GPCM_ERATTR0).....	1309
25.3.77	GPCM Transfer Error Attributes register 1 (IFC_GPCM_ERATTR1).....	1310
25.3.78	GPCM Transfer Error Attributes register 2 (IFC_GPCM_ERATTR2).....	1311
25.3.79	GPCM Status register (IFC_GPCM_STAT).....	1313
25.4	IFC functional description.....	1314
25.4.1	General architecture.....	1316
25.4.2	Programming model for flash interface timing .....	1324
25.4.3	Booting methods.....	1324
25.4.4	Software reset handling.....	1324
25.4.5	Data buffer control (BCTL).....	1325
25.4.6	External transceiver enable (TE).....	1327
25.5	NAND flash control machine.....	1329
25.5.1	NAND flash synchronous mode.....	1329
25.5.2	NAND flash asynchronous mode.....	1330
25.5.3	NV-DDR Mode.....	1330
25.5.4	Write protect.....	1330
25.5.5	SRAM buffer.....	1332
25.5.6	Use of ECC algorithms.....	1337

<b>Section number</b>	<b>Title</b>	<b>Page</b>
25.5.7	Programming the NAND FCM.....	1345
25.5.8	Looping FIR sequences.....	1349
25.5.9	NAND DLL.....	1349
25.5.10	NAND asynchronous mode timings.....	1350
25.5.11	NV-DDR mode timings.....	1354
25.5.12	NAND asynchronous mode boot mechanism.....	1356
25.6	NOR flash control machine.....	1360
25.6.1	NOR boot.....	1360
25.6.2	Unmuxed (parallel) asynchronous NOR read timings.....	1360
25.6.3	Simple asynchronous NOR write timings.....	1361
25.6.4	Muxed (ADM) asynchronous NOR read timings.....	1362
25.6.5	Muxed (ADM) asynchronous NOR write timings.....	1363
25.7	General purpose chip-select machine (GPCM).....	1364
25.7.1	Normal GPCM mode of operation.....	1364
25.7.2	Generic ASIC mode of operation.....	1376
25.8	Clock generation module.....	1380
25.9	Initialization/Application information.....	1380
25.9.1	Switching Interfaces in ONFi NAND.....	1380
25.9.2	ONFI mode timing parameters.....	1383
25.9.3	DLL configuration guideline (valid when IFC is in NVDDR Mode).....	1383
25.9.4	IFC flash connections.....	1384
25.9.5	Bus turnaround.....	1388
25.9.6	Interfacing to different port sizes.....	1389
25.9.7	Command sequence examples for NAND flash EEPROM.....	1390

## Chapter 26 Inter-Integrated Circuit (I2C)

26.1	The I2C module as implemented on the chip.....	1397
26.1.1	LS1043A I2C module integration.....	1397
26.1.2	LS1043A I2C module special consideration.....	1397

<b>Section number</b>	<b>Title</b>	<b>Page</b>
26.2	Overview.....	1398
26.3	Introduction to I2C.....	1398
26.3.1	Definition: I2C module.....	1399
26.3.2	Advantages of the I2C bus.....	1399
26.3.3	Module block diagram.....	1399
26.3.4	Features.....	1400
26.3.5	Modes of operation.....	1401
26.3.6	Definition: I2C conditions.....	1402
26.4	External signal descriptions.....	1403
26.4.1	Signal overview.....	1403
26.4.2	Detailed external signal descriptions.....	1403
26.5	Memory map and register definition.....	1403
26.5.1	Register accessibility.....	1404
26.5.2	Register figure conventions.....	1404
26.5.3	I2C Bus Address Register (I2Cx_IBAD).....	1405
26.5.4	I2C Bus Frequency Divider Register (I2Cx_IBFD).....	1406
26.5.5	I2C Bus Control Register (I2Cx_IBCR).....	1406
26.5.6	I2C Bus Status Register (I2Cx_IBSR).....	1408
26.5.7	I2C Bus Data I/O Register (I2Cx_IBDR).....	1409
26.5.8	I2C Bus Interrupt Config Register (I2Cx_IBIC).....	1410
26.6	Functional description.....	1411
26.6.1	Notes about module operation.....	1411
26.6.2	Transactions.....	1411
26.6.3	Arbitration procedure.....	1415
26.6.4	Clock behavior.....	1416
26.6.5	Interrupts.....	1424
26.6.6	STOP mode.....	1425
26.6.7	DMA interface.....	1425
26.7	Initialization/application information.....	1426

Section number	Title	Page
26.7.1	Recommended interrupt service flow.....	1426
26.7.2	General programming guidelines (for both master and slave mode).....	1427
26.7.3	Programming guidelines specific to master mode.....	1429
26.7.4	Programming guidelines specific to slave mode.....	1433
26.7.5	DMA application information.....	1433

## Chapter 27 Low Power Universal asynchronous receiver/transmitter (LPUART)

27.1	LPUART module integration.....	1441
27.2	Chip LPUART signals.....	1442
27.3	Chip LPUART module special consideration.....	1442
27.4	Introduction.....	1443
27.4.1	Features.....	1443
27.4.2	Modes of operation.....	1444
27.4.3	Signal Descriptions.....	1444
27.4.4	Block diagram.....	1444
27.5	Register definition.....	1446
27.5.1	LPUART Baud Rate Register (LPUARTx_BAUD).....	1448
27.5.2	LPUART Status Register (LPUARTx_STAT).....	1450
27.5.3	LPUART Control Register (LPUARTx_CTRL).....	1454
27.5.4	LPUART Data Register (LPUARTx_DATA).....	1458
27.5.5	LPUART Match Address Register (LPUARTx_MATCH).....	1459
27.5.6	LPUART Modem IrDA Register (LPUARTx_MODIR).....	1460
27.5.7	LPUART FIFO Register (LPUARTx_FIFO).....	1462
27.5.8	LPUART Watermark Register (LPUARTx_WATER).....	1464
27.6	Functional description.....	1465
27.6.1	Baud rate generation.....	1465
27.6.2	Transmitter functional description.....	1466
27.6.3	Receiver functional description.....	1469
27.6.4	Additional LPUART functions.....	1474

<b>Section number</b>	<b>Title</b>	<b>Page</b>
27.6.5	Infrared interface.....	1476
27.6.6	Interrupts and status flags.....	1477
<b>Chapter 28</b> <b>PCI Express Interface Controller</b>		
28.1	The PCI Express controller as implemented on the chip.....	1479
28.1.1	PCI Express MSI implementation.....	1479
28.1.2	PCI Express soft reset support.....	1481
28.1.3	PCI Express PM turnoff message support.....	1481
28.1.4	Additional PCI Express events connected to GIC-400 interrupt.....	1481
28.2	Introduction.....	1482
28.2.1	Overview.....	1482
28.2.2	Features.....	1485
28.2.3	Modes of Operation.....	1485
28.3	External Signal Descriptions.....	1486
28.4	Memory map/register overview.....	1487
28.4.1	PCI Express configuration registers.....	1487
28.4.2	PEX register descriptions.....	1488
28.5	PEX_LUT memory map/registers overview.....	1597
28.5.1	PEX_LUT register descriptions.....	1598
28.6	Functional Description.....	1609
28.6.1	Architecture.....	1610
28.6.2	Interrupts.....	1628
28.6.3	Initial Credit Advertisement.....	1628
28.6.4	Power Management.....	1628
28.6.5	Hot Reset.....	1629
28.7	Initialization/Application Information.....	1629
28.7.1	Configuring the chip for inbound CCSR accesses.....	1629
28.7.2	Poisoned TLP handling.....	1630

## Chapter 29

Section number	Title	Page
	<b>Pre-Boot Loader (PBL)</b>	
29.1	Overview.....	1631
29.2	Features summary.....	1632
29.3	Modes of operation.....	1632
29.4	Functional description.....	1633
29.4.1	Device configuration using reset configuration word (RCW).....	1633
29.4.2	Device initialization by PBL.....	1633
29.4.3	Required format of data structure used by PBL.....	1635
29.4.4	RCW loading by PBL.....	1637
29.4.5	Pre-boot initialization command loading by PBL.....	1638
29.4.6	Reserved address space used as internal PBL commands.....	1638
29.4.7	Error codes.....	1640
29.5	Initialization/Application information.....	1649
29.5.1	Starting addresses.....	1649
29.5.2	Software restrictions.....	1649
29.5.3	Software recommendations.....	1650

## Chapter 30 Quad Serial Peripheral Interface (QuadSPI)

30.1	The QuadSPI module as implemented on the chip.....	1651
30.1.1	LS1043A QuadSPI signals.....	1651
30.1.2	QuadSPI register reset values.....	1651
30.1.3	QuadSPI_MCR[SCLKCFG] mapping.....	1652
30.1.4	QuadSPI boot initialization sequence.....	1653
30.1.5	LS1043A QuadSPI module special consideration.....	1653
30.2	Introduction.....	1654
30.2.1	Features.....	1654
30.2.2	Block Diagram.....	1655
30.2.3	QuadSPI Modes of Operation.....	1656
30.2.4	Acronyms and Abbreviations.....	1656

<b>Section number</b>	<b>Title</b>	<b>Page</b>
30.2.5	Glossary for QuadSPI module.....	1657
30.3	External Signal Description.....	1658
30.3.1	Driving External Signals.....	1659
30.4	Memory Map and Register Definition.....	1661
30.4.1	Register Write Access.....	1661
30.4.2	Peripheral Bus Register Descriptions.....	1662
30.4.3	Serial Flash Address Assignment.....	1701
30.4.4	AMBA Bus Register Memory Map.....	1702
30.4.5	AHB Bus Register Memory Map Descriptions.....	1703
30.5	Interrupt Signals.....	1710
30.6	Functional Description.....	1710
30.6.1	Serial Flash Access Schemes.....	1710
30.6.2	Modes of Operation.....	1711
30.6.3	Normal Mode.....	1712
30.7	Initialization/Application Information.....	1730
30.7.1	Power Up and Reset.....	1730
30.7.2	Available Status/Flag Information.....	1730
30.7.3	Exclusive Access to Serial Flash for AHB Commands.....	1733
30.7.4	Command Arbitration .....	1734
30.7.5	Flash Device Selection.....	1735
30.7.6	DMA Usage.....	1735
30.7.7	Parallel mode.....	1737
30.8	Byte Ordering - Endianness.....	1739
30.8.1	Programming Flash Data.....	1740
30.8.2	Reading Flash Data into the RX Buffer.....	1741
30.8.3	Reading Flash Data into the AHB Buffer.....	1742
30.9	Serial Flash Devices.....	1742
30.9.1	Example Sequences.....	1742
30.9.2	Dual Die Flashes.....	1745

<b>Section number</b>	<b>Title</b>	<b>Page</b>
30.9.3	Boot initialization sequence.....	1746
30.9.4	Serial Flash Clock Frequency Limitations.....	1747
30.10	Sampling of Serial Flash Input Data.....	1747
30.10.1	Basic Description.....	1747
30.10.2	SDR mode.....	1748

## **Chapter 31** **Queue Direct Memory Access Controller (qDMA)**

31.1	Introduction .....	1751
31.1.1	Overview.....	1751
31.1.2	Features.....	1752
31.1.3	Modes of Operation.....	1753
31.2	Memory Map.....	1754
31.2.1	qDMA register descriptions.....	1754
31.3	Functional Description .....	1812
31.3.1	qDMA Command Queue Mode Operation.....	1812
31.3.2	qDMA Controller Interrupts.....	1826
31.3.3	Command Queue Interrupts.....	1826
31.3.4	Global Bandwidth Throttling.....	1827
31.3.5	Address Alignment Requirements.....	1828
31.3.6	Transaction Sizes.....	1828
31.3.7	Performance Improvements.....	1828
31.3.8	Low-Power Mode.....	1829
31.3.9	DMA Scenarios.....	1829
31.3.10	Error Handling.....	1830
31.4	Initialization Information.....	1835

## **Chapter 32** **SATA 3.0**

32.1	Advanced host controller interface overview .....	1837
32.2	SATA features summary.....	1838

<b>Section number</b>	<b>Title</b>	<b>Page</b>
32.3	SATA AHCI register descriptions.....	1838
32.3.1	SATA memory map.....	1839
32.3.2	HBA capabilities register (CAP).....	1840
32.3.3	Global HBA control register (GHC).....	1843
32.3.4	AHCI version register (VS).....	1845
32.3.5	Command completion coalescing control register (CCC_CTL).....	1846
32.3.6	HBA capabilities extended register (CAP2).....	1847
32.3.7	Port config register (PCFG).....	1848
32.3.8	Port Phy1Cfg register (PPCFG).....	1850
32.3.9	Port Phy2Cfg register (PP2C).....	1852
32.3.10	Port Phy3Cfg register (PP3C).....	1853
32.3.11	Port Phy4Cfg register (PP4C).....	1855
32.3.12	Port Phy5Cfg register (PP5C).....	1856
32.3.13	AXI cache control register (AXICC).....	1857
32.3.14	Port AXICfg register (PAXIC).....	1860
32.3.15	Port TransCfg register (PTC).....	1862
32.3.16	Port LinkCfg register (PLC).....	1863
32.3.17	Port LinkCfg1 register (PLC1).....	1865
32.3.18	Port LinkCfg2 register (PLC2).....	1866
32.3.19	Port LinkStatus1 register (PLS1).....	1867
32.3.20	Port CmdConfig register (PCMDC).....	1869
32.3.21	Port PhyControl status register (PPCS).....	1870
32.3.22	Timer control register (TCR).....	1871
32.3.23	Port x command list base address register (PxCLB).....	1872
32.3.24	Port x command list base address upper 32-bit register (PxCLBU).....	1873
32.3.25	Port x FIS base address register (PxFB).....	1874
32.3.26	Port x FIS base address upper 32-bit register (PxFBU).....	1875
32.3.27	Port x interrupt status register (PxIS).....	1876
32.3.28	Port x command and status register (PxCMD).....	1879

<b>Section number</b>	<b>Title</b>	<b>Page</b>
32.3.29	Port x SATA status register (PxSSTS).....	1883
32.3.30	Port x SATA control register (PxSCTL).....	1884
32.3.31	Port x SATA error register (PxSERR).....	1886
32.3.32	Port x command issue register (PxCI).....	1888
32.3.33	Port x FIS-based switching control register (PxFBS).....	1889
32.3.34	Port 0 BIST error register (PBERR).....	1891
32.4	Command layer.....	1892
32.4.1	Local port context management.....	1892
32.4.2	Vendor-specific BIST operation.....	1893
32.5	PhyControl BIST modes.....	1895
32.6	PHY control configuration register OOB timing setup.....	1897
32.7	Modifications to the AHCI standard PRDT entry.....	1898
32.8	Transport layer architectural overview.....	1899
32.9	Link layer overview.....	1900
32.9.1	General operation.....	1900
32.9.2	List of functions.....	1901
32.9.3	Link layer state machines.....	1901
32.9.4	Frame content scrambler and descrambler.....	1904
32.9.5	CRC generator and checker.....	1905
32.9.6	8B/10B encode and decode.....	1905
32.9.7	CONT primitive processing.....	1905
32.9.8	ALIGN insertion.....	1906
32.9.9	Debug functionality.....	1906
32.9.10	BIST support.....	1907
32.10	PHY control layer overview.....	1908
32.11	Reset.....	1908
32.11.1	Software reset.....	1909
32.11.2	Port reset.....	1912
32.11.3	HBA reset.....	1912

<b>Section number</b>	<b>Title</b>	<b>Page</b>
<b>Chapter 33 SerDes Module</b>		
33.1	The SerDes module as implemented on the chip.....	1915
33.1.1	SerDes lane assignments and multiplexing.....	1915
33.1.2	SerDes clocking.....	1918
33.2	Overview.....	1919
33.2.1	Features.....	1919
33.3	Modes of Operation.....	1920
33.4	External Signals Description.....	1920
33.5	SerDes register descriptions.....	1921
33.5.1	SerDes memory map.....	1921
33.5.2	SerDes PLLa Reset Control Register (PLL1RSTCTL - PLL2RSTCTL).....	1923
33.5.3	SerDes PLLa Control Register 0 (PLL1CR0 - PLL2CR0).....	1925
33.5.4	SerDes PLLa Control Register 1 (PLL1CR1 - PLL2CR1).....	1928
33.5.5	SerDes PLLa Control Register 5 (PLL1CR5 - PLL2CR5).....	1929
33.5.6	SerDes Transmit Calibration Control Register (TCALCR).....	1930
33.5.7	SerDes Transmit Calibration Control Register 1 (TCALCR1).....	1931
33.5.8	Receive Calibration Control Register (RCALCR).....	1933
33.5.9	SerDes Receive Calibration Control Register 1 (RCALCR1).....	1934
33.5.10	General Control Register 0 (GR0).....	1936
33.5.11	Lane a Protocol Select Status Register 0 (LNAPSSR0 - LNDPSSR0).....	1937
33.5.12	Protocol Configuration Register 0 (PCCR0).....	1939
33.5.13	Protocol Configuration Register 2 (PCCR2).....	1941
33.5.14	Protocol Configuration Register 8 (PCCR8).....	1942
33.5.15	Protocol Configuration Register 9 (PCCR9).....	1944
33.5.16	Protocol Configuration Register B (PCCRB).....	1946
33.5.17	General Control Register 0 - Lane a (LNAGCR0 - LNDGCR0).....	1948
33.5.18	General Control Register 1 - Lane a (LNAGCR1 - LNDGCR1).....	1951
33.5.19	Speed Switch Control Register 0 - Lane a (LNASSCR0 - LNDSSCR0).....	1955

<b>Section number</b>	<b>Title</b>	<b>Page</b>
33.5.20	Receive Equalization Control Register 0 - Lane a (LNARECR0 - LNDRECR0).....	1959
33.5.21	Receive Equalization Control Register 1- Lane a (LNARECR1 - LNDRECR1).....	1962
33.5.22	Transmit Equalization Control Register 0 - Lane a (LNATECR0 - LNDTECR0).....	1963
33.5.23	Speed Switch Control Register 1- Lane 0 (LNASSCR1 - LNDSSCR1).....	1966
33.5.24	TTL Control Register 0 - Lane a (LNATTLCR0 - LNDTTLCR0).....	1970
33.5.25	Test Control/Status Register 3 - Lane a (LNATCSR3 - LNDTCR3).....	1971
33.5.26	PEXa Protocol Control Register 0 (PEXACR0 - PEXCCR0).....	1973
33.5.27	SGMIIa Protocol Control Register 1 (SGMIIACR1 - SGMIIDCR1).....	1974
33.5.28	SGMIIa Protocol Control Register 3 (SGMIIACR3 - SGMIIDCR3).....	1976
33.5.29	QSGMIIa Protocol Control Register 1 (QSGMIIACR1 - QSGMIIIBCR1).....	1977
33.5.30	QSGMIIa Protocol Control Register 3 (QSGMIIACR3 - QSGMIIIBCR3).....	1978
33.5.31	XFIa Protocol Control Register 1 (XFIACR1 - XFIBCR1).....	1980
33.5.32	XFIa Protocol Control Register 3 (XFIACR3 - XFIBCR3).....	1981
33.6	MDIO register spaces.....	1983
33.6.1	MDIO_XFI_PMD register descriptions.....	1984
33.6.2	MDIO_XFI_PCS register descriptions.....	1991
33.6.3	MDIO_XFI_AN register descriptions.....	2018
33.6.4	MDIO_XFI_VENDOR_SPEC register descriptions.....	2041
33.6.5	MDIO_KX_PCS register descriptions.....	2053
33.6.6	MDIO_KX_AN register descriptions.....	2081
33.6.7	MDIO_KX_VENDOR_SPEC register descriptions.....	2106
33.6.8	MDIO_SGMII register descriptions.....	2113
33.6.9	MDIO_QSGMII register descriptions.....	2132
33.7	Initialization/Application Information.....	2149
33.7.1	Initialization.....	2150
33.7.2	Unused Lanes.....	2152
33.7.3	Soft Reset and Reconfiguring Procedures.....	2153
33.7.4	Quiesce Sequences for System Sleep.....	2154

## Chapter 34

Section number	Title	Page
	<b>Serial Peripheral Interface (SPI)</b>	
34.1	The SPI module as implemented on the chip.....	2155
34.1.1	LS1043A SPI signals.....	2155
34.1.2	LS1043A SPI module integration.....	2155
34.1.3	LS1043A SPI module special consideration.....	2156
34.2	Introduction.....	2156
34.2.1	Block Diagram.....	2156
34.2.2	Features.....	2157
34.2.3	Interface configurations.....	2159
34.2.4	Modes of Operation.....	2160
34.3	Module signal descriptions.....	2161
34.3.1	PCS0—Peripheral Chip Select.....	2161
34.3.2	PCS1–PCS3—Peripheral Chip Selects 1–3.....	2161
34.3.3	SCK—Serial Clock.....	2162
34.3.4	SIN—Serial Input.....	2162
34.3.5	SOUT—Serial Output.....	2162
34.4	Memory Map/Register Definition.....	2162
34.4.1	Module Configuration Register (SPI_MCR).....	2165
34.4.2	Transfer Count Register (SPI_TCR).....	2169
34.4.3	Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR $n$ ).....	2169
34.4.4	Status Register (SPI_SR).....	2174
34.4.5	DMA/Interrupt Request Select and Enable Register (SPI_RSER).....	2177
34.4.6	PUSH TX FIFO Register In Master Mode (SPI_PUSHR).....	2180
34.4.7	POP RX FIFO Register (SPI_POPR).....	2182
34.4.8	Transmit FIFO Registers (SPI_TXFR $n$ ).....	2183
34.4.9	Receive FIFO Registers (SPI_RXFR $n$ ).....	2184
34.4.10	Clock and Transfer Attributes Register Extended (SPI_CTARE $n$ ).....	2184
34.4.11	Status Register Extended (SPI_SREX).....	2186
34.5	Functional description.....	2187

<b>Section number</b>	<b>Title</b>	<b>Page</b>
34.5.1	Start and Stop of module transfers.....	2188
34.5.2	Serial Peripheral Interface SPI configuration.....	2189
34.5.3	Module baud rate and clock delay generation.....	2193
34.5.4	Transfer formats.....	2196
34.5.5	Continuous Serial Communications Clock.....	2202
34.5.6	Parity Generation and Check.....	2203
34.5.7	Interrupts/DMA requests.....	2204
34.5.8	Power saving features.....	2208
34.6	Initialization/application information.....	2209
34.6.1	How to manage queues.....	2209
34.6.2	Initializing Module in Master Mode.....	2210
34.6.3	Baud rate settings.....	2210
34.6.4	Delay settings.....	2211
34.6.5	Calculation of FIFO pointer addresses.....	2212

## **Chapter 35** **Thermal Monitoring Unit (TMU)**

35.1	The TMU module as implemented on the chip.....	2215
35.1.1	Local temperature sensor placement.....	2215
35.1.2	Initialization Information.....	2215
35.2	Thermal Monitoring Unit Introduction .....	2219
35.2.1	TMU Overview.....	2219
35.2.2	Features.....	2220
35.2.3	Modes of Operation.....	2221
35.3	TMU register descriptions.....	2221
35.3.1	TMU memory map.....	2221
35.3.2	TMU mode register (TMR).....	2222
35.3.3	TMU status register (TSR).....	2224
35.3.4	TMU monitor temperature measurement interval register (TMTMIR).....	2225
35.3.5	TMU interrupt enable register (TIER).....	2227

<b>Section number</b>	<b>Title</b>	<b>Page</b>
35.3.6	TMU interrupt detect register (TIDR).....	2228
35.3.7	TMU interrupt site capture register (TISCR).....	2230
35.3.8	TMU interrupt critical site capture register (TICSCR).....	2231
35.3.9	TMU monitor high temperature capture register (TMHTCR).....	2232
35.3.10	TMU monitor low temperature capture register (TMLTCR).....	2233
35.3.11	TMU monitor high temperature immediate threshold register (TMHTITR).....	2234
35.3.12	TMU monitor high temperature average threshold register (TMHTATR).....	2235
35.3.13	TMU monitor high temperature average critical threshold register (TMHTACTR).....	2237
35.3.14	TMU temperature configuration register (TTCFGR).....	2238
35.3.15	TMU sensor configuration register (TSCFGR).....	2239
35.3.16	TMU report immediate temperature site register a (TRITSR0 - TRITSR4).....	2240
35.3.17	TMU report average temperature site register a (TRATSR0 - TRATSR4).....	2241
35.3.18	TMU temperature range 0 control register (TTR0CR).....	2242
35.3.19	TMU temperature range 1 control register (TTR1CR).....	2244
35.3.20	TMU temperature range 2 control register (TTR2CR).....	2246
35.3.21	TMU temperature range 3 control register (TTR3CR).....	2247
35.4	Functional Description.....	2249
35.4.1	Monitoring.....	2249
35.4.2	Reporting.....	2250

## **Chapter 36** **Universal Serial Bus Interface 3.0**

36.1	Overview.....	2251
36.1.1	Features.....	2252
36.1.2	Modes of Operation.....	2252
36.1.3	External Signals.....	2252
36.2	USB Memory Map/Register Definition.....	2253
36.2.1	Capability Registers Length and HC Interface Version Number (USBx_CAPLENGTH).....	2268
36.2.2	Host Controller Structural Parameters 1 (USBx_HCSPARAMS1).....	2269
36.2.3	Host Controller Structural Parameters 2 (USBx_HCSPARAMS2).....	2269

<b>Section number</b>	<b>Title</b>	<b>Page</b>
36.2.4	Host Controller Structural Parameters 3 (USBx_HCSPARAMS3).....	2270
36.2.5	Host Controller Capability Parameters 1 (USBx_HCCPARAMS1).....	2271
36.2.6	Doorbell Offset (USBx_DBOFF).....	2272
36.2.7	Runtime Register Space Offset (USBx_RTSOFF).....	2273
36.2.8	Host Controller Capability Parameters 2 (USBx_HCCPARAMS2).....	2273
36.2.9	Global SoC Bus Configuration Register 0 (USBx_GSBUSCFG0).....	2274
36.2.10	Global SoC Bus Configuration Register 1 (USBx_GSBUSCFG1).....	2279
36.2.11	Global Tx Threshold Control Register (USBx_GTXTHRCFG).....	2280
36.2.12	Global Rx Threshold Control Register (USBx_GRXTHRCFG).....	2281
36.2.13	Global Core Control Register (USBx_GCTL).....	2283
36.2.14	Global Status Register (USBx_GSTS).....	2288
36.2.15	Global User Control Register 1 (USBx_GUCTL1).....	2290
36.2.16	Global User ID Register (USBx_GUID).....	2294
36.2.17	Global User Control Register (USBx_GUCTL).....	2295
36.2.18	Global SoC Bus Error Address Register low (USBx_GBUSERRADDRLO).....	2298
36.2.19	Global SoC Bus Error Address Register high (USBx_GBUSERRADDRHI).....	2299
36.2.20	Global SS Port to Bus Instance Mapping Register - Low (USBx_GPRTBIMAPLO).....	2299
36.2.21	Global SS Port to Bus Instance Mapping Register - High (USBx_GPRTBIMAPHI).....	2300
36.2.22	Global Hardware Parameters Register 0 (USBx_GHWPARAMS0).....	2300
36.2.23	Global Hardware Parameters Register 1 (USBx_GHWPARAMS1).....	2302
36.2.24	Global Hardware Parameters Register 2 (USBx_GHWPARAMS2).....	2305
36.2.25	Global Hardware Parameters Register 3 (USBx_GHWPARAMS3).....	2306
36.2.26	Global Hardware Parameters Register 4 (USBx_GHWPARAMS4).....	2309
36.2.27	Global Hardware Parameters Register 5 (USBx_GHWPARAMS5).....	2311
36.2.28	Global Hardware Parameters Register 6 (USBx_GHWPARAMS6).....	2312
36.2.29	Global Hardware Parameters Register 7 (USBx_GHWPARAMS7).....	2314
36.2.30	Global High-Speed Port to Bus Instance Mapping Register - Low (USBx_GPRTBIMAP_HSLO).....	2315
36.2.31	Global High-Speed Port to Bus Instance Mapping Register - High (USBx_GPRTBIMAP_HSHI).....	2315
36.2.32	Global USB2 PHY Configuration Register (USBx_GUSB2PHYCFG).....	2316

<b>Section number</b>	<b>Title</b>	<b>Page</b>
36.2.33	Global USB 3.0 PIPE Control Register (USB <sub>x</sub> _GUSB3PIPECTL).....	2319
36.2.34	Global Transmit FIFO Size Register (USB <sub>x</sub> _GTXFIFOSIZ <sub>n</sub> ).....	2321
36.2.35	Global Receive FIFO Size Register (USB <sub>x</sub> _GRXFIFOSIZ <sub>n</sub> ).....	2323
36.2.36	Global Event Buffer Address (Low) Register (USB <sub>x</sub> _GEVNTADRLO).....	2323
36.2.37	Global Event Buffer Address (High) Register (USB <sub>x</sub> _GEVNTADRHI).....	2324
36.2.38	Global Event Buffer Size Register (USB <sub>x</sub> _GEVNTSIZ).....	2325
36.2.39	Global Event Buffer Count Register (USB <sub>x</sub> _GEVNTCOUNT).....	2326
36.2.40	Global Hardware Parameters Register 8 (USB <sub>x</sub> _GHWPARAMS8).....	2326
36.2.41	Global Device TX FIFO DMA Priority Register (USB <sub>x</sub> _GTXFIFOPRIDEV).....	2327
36.2.42	Global Host TX FIFO DMA Priority Register (USB <sub>x</sub> _GRXFIFOPRIHST).....	2328
36.2.43	Global Host RX FIFO DMA Priority Register (USB <sub>x</sub> _GRXFIFOPRIHST).....	2329
36.2.44	Global Host FIFO DMA High-Low Priority Ratio Register (USB <sub>x</sub> _GDMAHLRATIO).....	2330
36.2.45	Global Frame Length Adjustment Register (USB <sub>x</sub> _GFLADJ).....	2331
36.2.46	Device Configuration Register (USB <sub>x</sub> _DCFG).....	2332
36.2.47	Device Control Register (USB <sub>x</sub> _DCTL).....	2334
36.2.48	Device Event Enable Register (USB <sub>x</sub> _DEVTEN).....	2339
36.2.49	Device Status Register (USB <sub>x</sub> _DSTS).....	2341
36.2.50	Device Generic Command Parameter Register (USB <sub>x</sub> _DGCMDPAR).....	2344
36.2.51	Device Generic Command Register (USB <sub>x</sub> _DGCMD).....	2344
36.2.52	Device Active USB Endpoint Enable Register (USB <sub>x</sub> _DALEPENA).....	2347
36.2.53	Device Physical Endpoint-n Command Parameter 2 Register (USB <sub>x</sub> _DEPCMDPAR2 <sub>n</sub> ).....	2348
36.2.54	Device Physical Endpoint-n Command Parameter 1 Register (USB <sub>x</sub> _DEPCMDPAR1 <sub>n</sub> ).....	2349
36.2.55	Device Physical Endpoint-n Command Parameter 0 Register (USB <sub>x</sub> _DEPCMDPAR0 <sub>n</sub> ).....	2349
36.2.56	Device Physical Endpoint-n Command Register (USB <sub>x</sub> _DEPCMD <sub>n</sub> ).....	2350
36.2.57	OTG Configuration Register (USB <sub>x</sub> _OCFG).....	2352
36.2.58	OTG Control Register (USB <sub>x</sub> _OCTL).....	2354
36.2.59	OTG Events Register (USB <sub>x</sub> _OEVT).....	2357
36.2.60	OTG Events Enable Register (USB <sub>x</sub> _OEVTEM).....	2361
36.2.61	OTG Status Register (USB <sub>x</sub> _OSTS).....	2364

<b>Section number</b>	<b>Title</b>	<b>Page</b>
36.2.62	ADP Configuration Register (USBx_ADPCFG).....	2366
36.2.63	ADP Control Register (USBx_ADPCCTL).....	2367
36.2.64	ADP Event Register (USBx_ADPEVT).....	2369
36.2.65	ADP Event Enable Register (USBx_ADPEVTCN).....	2371
36.3	USB PHY Memory Map/Register Definition.....	2371
36.3.1	SUP_IDCODE_LO (USB_PHY_SSx_IP_IDCODE_LO).....	2372
36.3.2	SUP_IDCODE_HI (USB_PHY_SSx_SUP_IDCODE_HI).....	2372
36.3.3	MPLL_LOOP_CTL (USB_PHY_SSx_MPLL_LOOP_CTL).....	2373
36.3.4	LANE0_RX_OVRD_IN_HI (USB_PHY_SSx_LANE0_RX_OVRD_IN_HI).....	2373
36.4	Functional Description.....	2374
36.4.1	System memory descriptor and data buffers.....	2374
36.4.2	Device descriptor structures.....	2375
36.4.3	Device Programming Model.....	2392
36.4.4	Host programming model.....	2436
36.4.5	Device physical endpoint-specific commands.....	2439
36.4.6	OTG.....	2447
36.4.7	Initialization/application information.....	2475
36.4.8	Power management overview.....	2475

## **Chapter 37 Watchdog Timer (WDOG)**

37.1	Overview.....	2479
37.1.1	Features.....	2480
37.2	Clocks.....	2481
37.3	Functional description.....	2481
37.3.1	Timeout event.....	2481
37.3.2	Interrupt event .....	2482
37.3.3	Operations.....	2482
37.3.4	Interrupt.....	2483
37.3.5	Flow Diagrams.....	2483

<b>Section number</b>	<b>Title</b>	<b>Page</b>
37.4	Initialization.....	2485
37.5	WDOG Memory Map/Register Definition.....	2486
37.5.1	Watchdog Control Register (WDOG <sub>x</sub> _WCR).....	2486
37.5.2	Watchdog Service Register (WDOG <sub>x</sub> _WSR).....	2488
37.5.3	Watchdog Reset Status Register (WDOG <sub>x</sub> _WRSR).....	2488
37.5.4	Watchdog Interrupt Control Register (WDOG <sub>x</sub> _WICR).....	2489

# Chapter 1 Overview

## 1.1 Introduction

The LS1043A QorIQ advanced multicore processor combines two to four Arm® Cortex®-v8 A53 cores with datapath acceleration optimized for L2/3 packet processing, single pass security offload, robust traffic management, and quality of service.

This advanced quad-core 64-bit Arm processor is ideal for applications such as, branch and enterprise routers, switches, firewall, packet filtering processors, and general-purpose embedded computing applications. The high level of integration delivers significant performance benefits, such as 10 GbE, multiple USB 3.0 interfaces, single source clock.

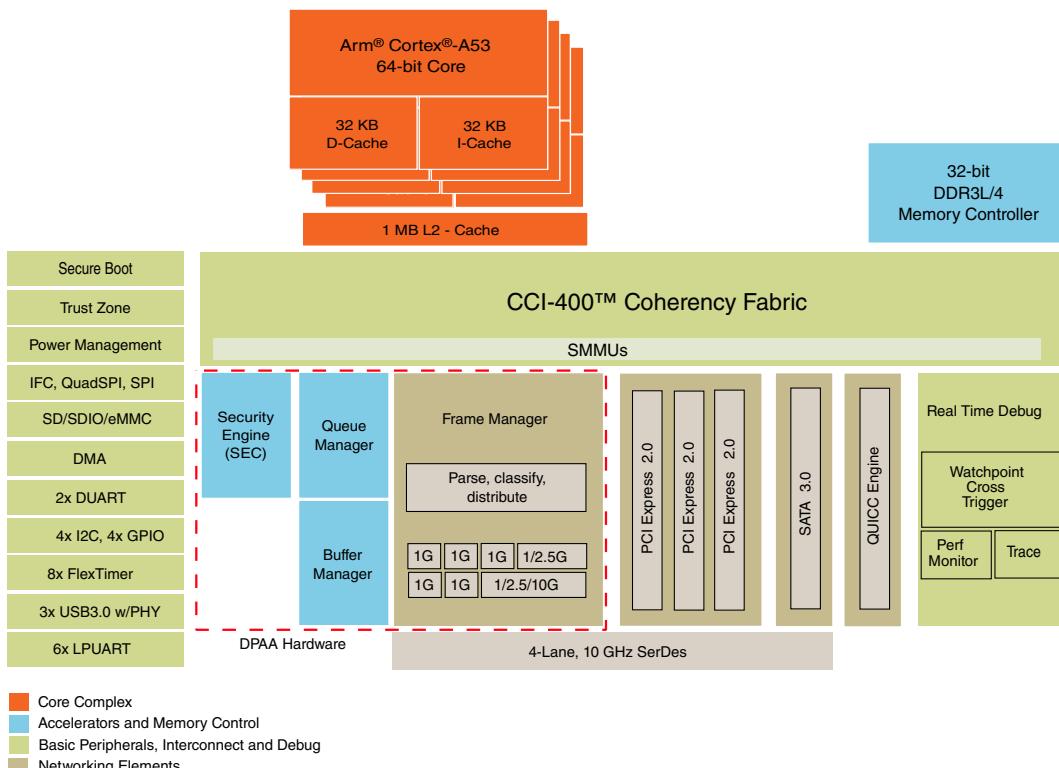


Figure 1-1. Block diagram

## 1.2 Features summary

This chip includes the following distinctive functions and features:

- Arm® Cortex®-v8 A53 (64-bit) with the following capabilities:
  - Speed up to 1.6 GHz
  - 32 KB L1 instruction cache and data cache for each core (ECC protection)
  - Neon SIMD co-processor
  - Arm v8 cryptography extensions
- 1 MB unified I/D L2 cache (ECC protection)
- Hierarchical interconnect fabric
  - Hardware managed data coherency
  - Up to 400 MHz operation
- One 32-bit DDR3L/DDR4 SDRAM memory controller
  - ECC and interleaving support
  - Up to 1.6 GT/s
- Data path acceleration architecture (DPAA) incorporating acceleration for the following functions:
  - Packet parsing, classification, and distribution (FMan)
  - Queue management for scheduling, packet sequencing, and congestion management (QMan)
  - Hardware buffer management for buffer allocation and de-allocation (BMan)
  - Cryptography acceleration (SEC)
- Parallel Ethernet interfaces
  - Up to two RGMII interfaces
- Four SerDes lanes for high-speed peripheral interfaces
  - Three PCI Express controllers, supporting x4 operation
  - One Serial ATA (SATA 3.0) controller
  - Up to four SGMII supporting 1000 Mbps
  - Up to two SGMII supporting 2500 Mbps
  - Up to one XFI (10GbE) interface
  - Up to one QSGMII
  - Supports 1000Base-KX
  - IEEE® 1588 support
  - Supports full-duplex mode only
- Additional peripheral interfaces:
  - One quad serial peripheral interface (QuadSPI) controller and one serial peripheral interface (SPI) controller
  - Integrated flash controller (IFC) supporting NAND and NOR flash with 28-bit addressing and 16-bit data interface

- Three USB 3.0 controllers with integrated PHY
- One enhanced secure digital host controller (eSDHC) supporting SD 3.0, eMMC 4.4 and eMMC 4.5 modes
- One QUICC engine (QE) block supporting TDM/HDLC
- Four I2C controllers
- Two 16550 compliant DUARTs, and six low power UARTs (LPUART)
- Four general-purpose I/O (GPIO)
- Eight FlexTimers/PWMs
- Five Watchdog timer
- Trust Architecture
- Debug supporting run control, data acquisition, high-speed trace, and performance/event monitoring

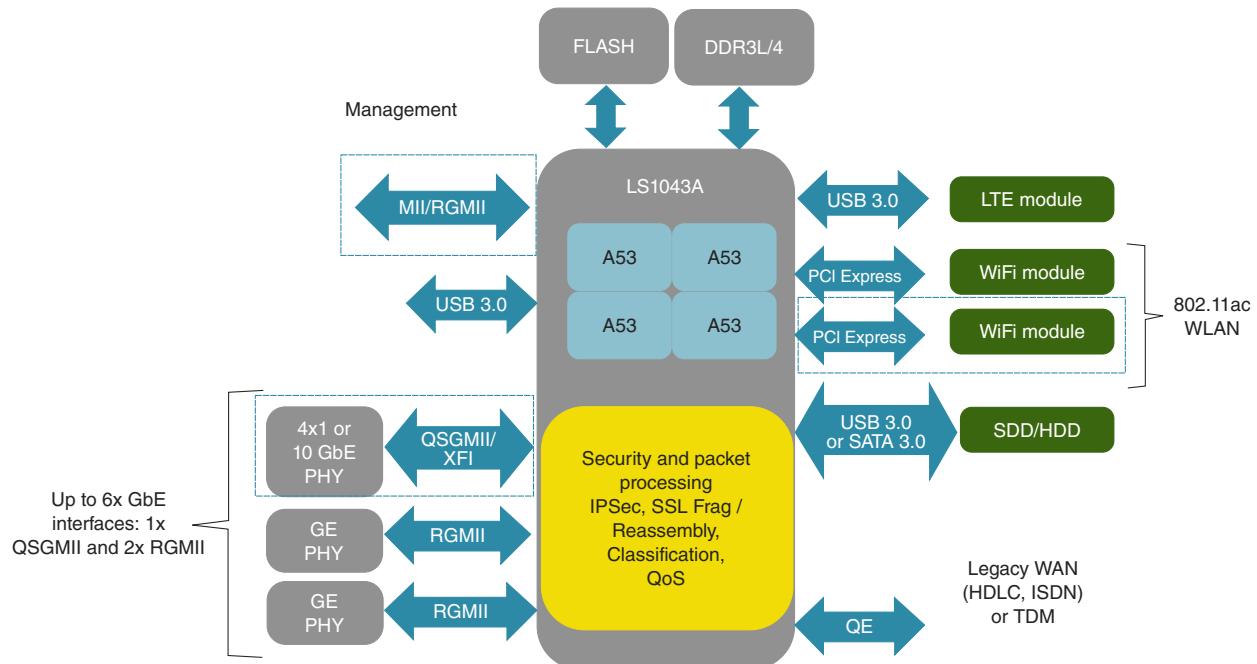
## 1.3 Application examples

The LS1043A core processors are very flexible and can be configured to meet many system application needs. The cores can be configured to be used in either a symmetric or asymmetric multiprocessing modes. For example, cores configured in asymmetric multiprocessing mode can be used in the case where each core runs operating systems independently, or in the case where each core performs separate tasks. This flexibility enables application developers to assign distinct processing resources to distinct tasks that need guaranteed performance.

The value proposition of this chip is further enhanced by a high degree of peripheral integration of system controllers such as, DDR4 controller, data path acceleration architecture (DPAA). The DPAA offloads packet parsing, classification, traffic management, and quality of service.

### 1.3.1 Multi-service branch office router

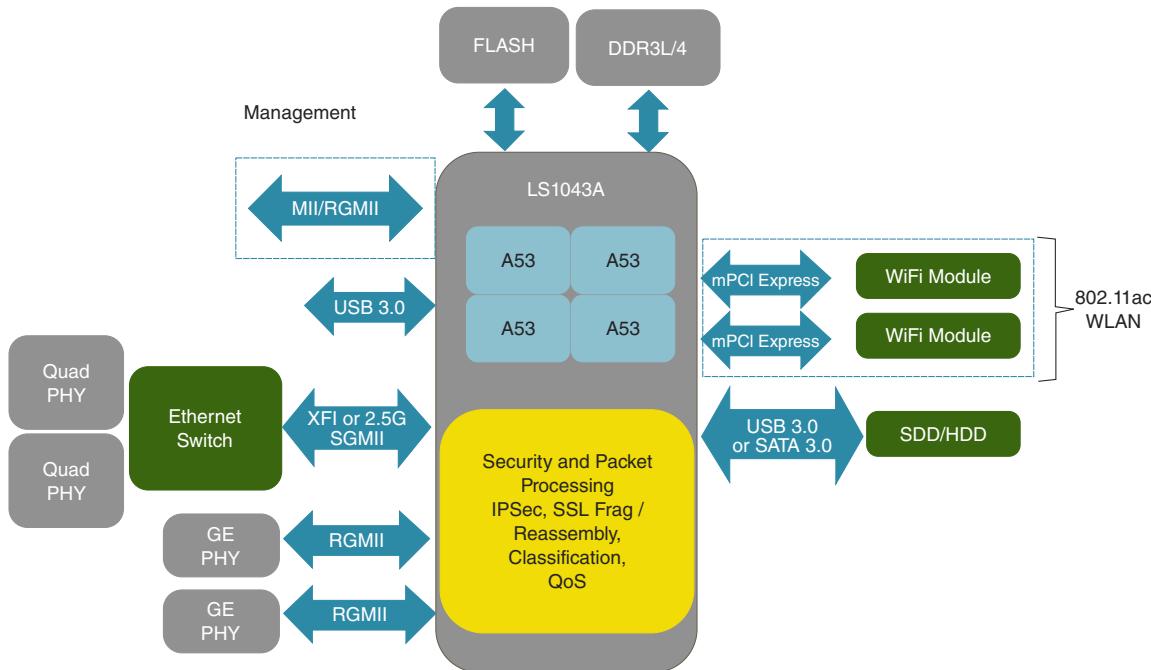
The chip allows for a low cost, feature dense, passively cooled branch office router design, which features hardware processing for advanced data path off load and has quad-core Arm 64-bit processors for high touch data and control path applications. It also supports hardware enabled virtualization elements for 3rd party applications. LS1043A also delivers the CPU bandwidth needed for next generation wired and wireless virtualized router platforms supporting SDN and NFV applications while maintaining high performance VPN links to the cloud.



**Figure 1-2. Multi-service branch office router**

### 1.3.2 Security appliance/UTM

In the security appliance space the chip comes with the option for 5 Gbps single pass cryptographic offload and 10 Gbps data path parse, classification and distribution which helps in delivering flows to cores for additional security processing. With quad-core design, the chip can run deep packet inspection, SPI firewall and IDS/IPS within a single low cost chip.



**Figure 1-3. Security appliance/UTM application diagram**

## 1.4 Module features

This section contains a high level view of the chip architecture.

### 1.4.1 Arm® Cortex®-A53 core

The Arm® Cortex®-A53 processor is an extremely power efficient Armv8 processor capable of supporting 32-bit and 64-bit code seamlessly. It makes use of a highly efficient 8-stage in-order pipeline balanced with advanced fetch and data access techniques for performance.

The multicore processing provides the ability for any of the four component processors, within a cluster, to shut down when not in use, for instance when the chip is in standby mode, to save power. When higher performance is required, every processor is in use to meet the demand while still sharing the workload to keep power consumption as low as possible.

The LS1043A features four high-performance Cortex A53 cores:

- 64 and 32-bit execution states for scalable high performance

- Multiple coherent SMP processor clusters through AMBA® 4 technology
- New instruction set, A64
- In-order pipeline with symmetric dual-issue of most instructions
- 32KB instruction cache, 32 KB data cache, 1MB unified L2 cache.
- NEON technology - Accelerates multimedia and signal processing algorithms such as video encode/decode, 2D/3D graphics, gaming, audio and speech processing, image processing, telephony, and sound synthesis. Also useful in accelerating floating point code with SIMD execution.
- Hardware-accelerated cryptography - 3x-10x better software encryption performance Useful for small granule decrypt/encrypt too small to efficiently offload to HW accelerator
- Floating point unit - Hardware support for floating point operations in half-, single- and double-precision floating point arithmetic. IEE754-2008 enhancements are included.
- TrustZone® Technology - Ensures reliable implementation of security applications ranging from digital rights management to electronic payment.
- Double Precision Floating Point SIMD - Allows SIMD vectorisation to be applied to a much wider set of algorithms (for example scientific / High Performance Computing (HPC) and supercomputer).
- 64-bit Virtual address reach - Enables virtual memory beyond 4GB 32b limit. Important for modern desktop and server software using memory mapped file I/O, sparse addressing.
- Enhanced Cache management - User space cache operations improve dynamic code generation efficiency, Data Cache Zero for fast clear.
- Extensive power-saving features - Hierarchical clock gating, power domains, advanced retention modes
- Hardware virtualization support
- NEON SIMD extensions onboard (per core)

## 1.4.2 System memory management unit MMU-500

The MMU-500 is a system-level memory management Unit (MMU) that translates an input address to an output address, by performing one or more translation table walks.

It supports the translation table formats defined by the Arm architecture, and can perform

- Stage 1 translations that translate an input Virtual Address (VA) to an output Physical Address (PA) or Intermediate Physical Address (IPA).
- Stage 2 translations that translate an input IPA to an output PA.
- Combined stage 1 and stage 2 translations that translate an input VA to an output IPA, and then translate that IPA to a PA. The MMU-500 performs a translation table walk for each stage of the translation.

The purpose of separating the process into 2 stages is to allow the guest OS to control the address translation between VA and what it “thinks” is the PA, while the hypervisor controls the translation from IPA to PA. This split process allows the hypervisor to separate the resources of different Virtual Machines (VMs). Address translation is performed both in the cores and also for IO devices, using the SMMU.

The MMU-500 is a distributed SMMU, which makes use of one central controller (TCU) and up to 32 translation units (TBUs). In this chip there is a single TCU, supporting a total of 4 TBUs.

### 1.4.3 Arm CoreLink CCI-400 cache coherent interconnect

The CCI-400 combines interconnect and coherency functions into a single module. The CCI-400 cache coherent interconnect is an infrastructure component that supports:

- Data coherency between both Cortex-A53 cores and all I/O masters with three independent Points-of-Serialization (PoS) and full barrier support high-bandwidth, cross-bar interconnect functionality between the masters and up to three slaves
- DVM message transport between masters
- Quality-of-Service (QoS) regulation for shaping traffic profiles
- Performance monitoring unit (PMU) to count performance-related events
- Programmers view (PV) to control the coherency and interconnect functionality

### 1.4.4 PreBoot loader (PBL) and nonvolatile memory interfaces

The PBL functions include the following:

- Simplifies boot operations, replacing pin strapping resistors with configuration data loaded from nonvolatile memory
- Uses the configuration data to initialize other system logic and to copy data from low speed memory interfaces IFC, QuadSPI, and SD/eSDHC/eMMC) into fully initialized DDR or OCRAMs.
- Releases CPU 0 from reset, allowing the boot processes to begin from fast system memory

The nonvolatile memory interfaces accessible by the PBL are described in the subsequent sections. These interfaces may be accessed by software running on the CPUs. these are not dedicated to the PBL. Note that the integrated flash controller (IFC) can be used for both volatile (SRAM) and nonvolatile memory, as well as a control and low performance data port for external memory-mapped devices.

### 1.4.5 DDR memory controllers

The DDR memory controller supports DDR3L and DDR4 SDRAM. The memory interface controls main memory accesses and supports a maximum of 32 GB of main memory. The chip supports chip-select interleaving within the controller.

The DDR memory controller can be configured to retain the currently active SDRAM page for pipe-lined burst accesses. Page mode support of up to 32 simultaneously open pages can dramatically reduce access latencies for page hits. Using ECC, the chip detects and corrects all single-bit errors and detects all double-bit errors and all errors within a nibble.

In addition, the DDR controller offers an initialization bypass feature for use by system designers to prevent reinitialization of main memory during system power-on after an abnormal shutdown. The DDR controller also supports active zeroization of system memory upon detection of a user-defined security violation.

### 1.4.6 Enhanced direct memory access (eDMA) and direct memory access multiplexer (DMAMUX)

It has the following general features:

- 32 channels support independent 8-, 16- or 32-bit single value or block transfers
- Supports variable sized queues and circular queues
- Source and destination address registers are independently configured to post increment or remain constant
- Each transfer is initiated by a peripheral, CPU, periodic timer interrupt or eDMA channel request
- Each DMA channel can optionally send an interrupt request to the CPU on completion of a single value or block transfer
- DMA transfers possible between system memories, General Purpose I/Os (GPIOs) and Slave Peripherals that support DMA
- Programmable DMAMUX allows assignment of any DMA source to any available DMA channel with up to a total of 64 potential request sources

The DMA channel request can be initiated by the following peripherals:

- 4x I<sup>2</sup>C
- 6x LPUART
- 1x SPI
- QuadSPI

The DMA requests from these peripherals are connected to eDMA through DMAMUX and the implementation details can be found in [LS1043A DMAMUX module special consideration](#).

The DMAMUX selects from many DMA requests down to 16 for the DMA controller. There are 2 DMAMUXs associated with each 32-channel DMA.

## 1.4.7 DUART

The DUART supports full-duplex operation and is compatible with the PC16450 and PC16550 programming models. All the transmitter and receiver support 16-byte FIFOs.

## 1.4.8 FlexTimer module (FTM)

The key features of the FTM are as follows:

- Selectable FTM source clock and programmable prescaler
- 16-bit counter supporting free-running or initial/final value and counting is up or up-down
- Input capture, output compare, edge aligned and center aligned PWM modes
- Operation of FTM channels as pairs with equal outputs, pairs with complimentary outputs, or independent channels with independent outputs
- Deadtime insertion is available for each complementary pair
- Generation of hardware triggers
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- Configurable channel polarity
- Programmable interrupt on input capture, reference compare, overflowed counter or detected fault condition
- Quadrature decoder with input filters, relative position counting and interrupt on position count or capture of position count on external event
- DMA support for FTM event

## 1.4.9 Integrated flash controller (IFC)

The integrated flash controller (IFC) is used to interface with external asynchronous NAND flash, asynchronous NOR flash, SRAM, generic ASIC memories, and EPROM. The IFC has the following general features:

## Module features

- Flash controller with seven chip-selects
- Functional multiplexing of pins between NAND, NOR, and GPCM
- Supports memory banks of sizes up to 256 MB (for NOR and GPCM)
- Write-protection capability for NAND and NOR devices
- External transceiver enable/disable control on per-bank basis

### 1.4.9.1 IFC NAND flash controller features

The IFC NAND Flash controller has the following features:

- Support for x8/x16 SLC/MLC NAND flash devices with page sizes up to 8 KB
- Support for ONFI-2.2 asynchronous interface (8-/16-bit)/NVDDR interface and mandatory commands
- 4-/8-/24-/40-bit ECC generation/checking
- Flexible timing control to allow interfacing with proprietary NAND devices
- Boot chip-select (CS0) available for NAND Flash at system reset
- Execute-in-place boot loading from NAND Flash (in asynchronous mode)

### 1.4.9.2 IFC NOR flash controller features

The IFC NOR Flash controller has the following features:

- Supports x8/x16 asynchronous NOR Flash devices
- Supports address data multiplexed (ADM) NOR devices
- Flexible timing control allows interfacing with proprietary NOR devices
- Boot chip-select (CS0) available for NOR flash at system reset

### 1.4.9.3 IFC GPCM controller features

The IFC general-purpose, chip-select controller (GPCM) supports operation in either normal or generic ASIC modes. Normal mode operation has the following features:

- Support for x8/x16-bit devices
- Compatible with general-purpose, addressable device such as SRAM and ROMs
- External clock is supported with programmable division ratio
- Output enable signal (OE)
- Byte-write enable signals
- Even/Odd parity on data bus
- External access termination signal
- Burst support in GPCM

General ASIC mode operation has the following features:

- Support for x8/x16 bit devices with shared address/data bus using the following address and data sequences:
  - 16 bit I/O: AADD
  - 8 bit I/O : AAAADDDD
- Supports configurable even/odd parity and parity error detection on address/data bus

### **1.4.10 Universal Serial Bus (USB) controllers and PHY**

The USB3.0 controllers provides point-to-point connectivity conforming to the Universal Serial Bus revision 3.0 specification. The USB controller and integrated PHY can be configured to operate as a stand-alone host, stand-alone device, or with both host and device functions operating simultaneously.

The host and device functions are configured to support the following types of USB transfers:

- Bulk
- Control
- Interrupt
- Isochronous

Key features of the USB controller include the following:

- OTG 2.0
- USB dual-role operation and can be configured as host or device
- Operation as a stand-alone USB device
  - One upstream facing port
  - Four bidirectional programmable USB endpoints
- Operation as a stand-alone USB host controller
  - USB root hub with one downstream-facing port
  - Enhanced host controller interface (EHCI) compatible
- Super-speed (5 GT/s), High-speed (480 Mbps), and full-speed (12 Mbps) operations.

### **1.4.11 High speed I/O interfaces**

The chip supports the SGMII, PCI Express 2.0 controller, and SATA high-speed I/O interface standards.

### 1.4.11.1 Serial ATA (SATA) controller

The SATA controller is compliant with the Serial ATA 3.0 Specification. The SATA controller has the following features:

- Supports speeds: 1.5 Gbps (first-generation SATA), 3 Gbps (second-generation SATA), and 6 Gbps (third-generation SATA)
- Single SATA 3.0 controller with chip-level interface
- Supports asynchronous notification and hot-plug

#### **NOTE**

This hot-plug capabilities are supported at generation 1 and generation 2 speeds.

- Asynchronous signal recovery
- Link power management
- Native command queuing
- Staggered spin-up
- Port multiplier support
- Standard ATA master-only emulation
- Contain ATA shadow registers
- SATA superset registers
- SError, SControl, SStatus
- Interrupt driven
- Contains Power management support
- Supports error handling and diagnostic features
- Far-end/near-end loopback
- Failed CRC error reporting
- Increased ALIGN insertion rates
- Scrambling and CONT override

### 1.4.11.2 PCI Express

Each of the three PCI Express is compatible with the PCI Express Base Specification Revision 3.0. Key features of the PCI Express interface include the following:

- Power-on reset configuration options allow root complex functionality
- The physical layer operates at 2.5 or 5 Gbps data rate per lane
- Receive and transmit ports operate independently, with an aggregate theoretical bandwidth of 32 Gbps
- x4, x2, and x1 link widths support
- Both 32- and 40-bit addressing and 256-byte maximum payload size
- Full 64-bit decode with 36-bit wide windows

- Inbound INTx transactions
- Message Signaled Interrupt (MSI) transactions

### 1.4.11.3 SGMII

The serial gigabit media independent interface (SGMII) is a high-speed interface linking the Ethernet controller with an Ethernet PHY. SGMII uses differential signaling for electrical robustness. Only four signals are required: receive data and its inverse, and send data and its inverse; no clock signals are required.

### 1.4.12 QUICC Engine (QE)

Single 32-bit RISC controller for flexible support of communications peripherals

- Serial DMA channel for reception and transmission on all serial channels
- Two UCCs supporting the following interfaces (not all of them simultaneously):
  - Serial
  - UART
  - Asynchronous HDLC (256 Channels) (bit rate up to 2 Mbps)
  - TDM interfaces supporting up to 128 QUICC multichannel controller channels

### 1.4.13 Enhanced secure digital host controller and SDIO

Detailed features of the SD/eSDHC/eMMC controller include the following:

- Conforms to the SD host controller standard specification version 3.0
- Compatible with the MMC system specification version 4.5
- Compatible with the SD memory card physical layer specification version 3.01
- Compatible with the SD - SDIO card specification version 2.0
- Designed to work with eMMC devices as well as SD memory, SDIO, and SD combo cards and their variants
- Supports SD UHS-1 speed modes

### 1.4.14 Security Engine (SEC)

The SEC is the chip's security engine, which serves as the latest cryptographic acceleration and offloading hardware. It combines functions previously implemented in separate modules to create a modular and scalable acceleration and assurance engine. It

also implements block encryption algorithms, stream cipher algorithms, hashing algorithms, public key algorithms, run-time integrity checking, and a hardware random number generator. SEC performs higher-level cryptographic operations than previous cryptographic accelerators. This provides significant improvement to system level performance. SEC includes the following interfaces:

- A slave bus interface for the processor to write configuration and command information, and to read status information
- A bus master interface that allows SEC to read/write data from external memory
- A Queue Manager interface that allows SEC to accept jobs directly from the Queue Manager module

The SEC provides the following functions:

- DMA for bus master operation
- Job Queue Controller with four Job Rings
- Descriptor Controllers (DECOs)
  - Responsible for executing descriptors and managing sequencing of keys, context, and data through the various CHAs
  - Performs header and trailer processing as defined by the descriptor
- Run-Time Integrity Checker (RTIC)
- Crypto Hardware Accelerators (CHAs)
  - Public Key Hardware Accelerator (PKHA)
  - Random Number Generator
  - Advanced Encryption Standard Accelerator (AES)
  - Message Digest Hardware Accelerator (MDHA)
  - SNOW 3G f8 Hardware Accelerator (SNOW f8)
  - SNOW 3G f9 Hardware Accelerator (SNOW f9)
  - ZUC Encryption Accelerator (ZUCE)
  - ZUC Authentication Accelerator (ZUCA)
  - Data Encryption Standard Hardware Accelerator (DESA)
  - Cyclic-Redundancy Check Accelerator (CRCA)
  - Kasumi f8 and f9 Hardware Accelerator (KFHA)

### **1.4.15 Inter-Integrated Circuit (I2C)**

The I2C allows communication between a number of devices.

## 1.4.16 Low Power Universal asynchronous receiver/ transmitter (LPUART)

The chip supports the asynchronous serial bus communication interface with programmable 8- or 9-bit data format and support of CEA709.1-B (LON), ISO 7816 smart card interface.

## 1.4.17 Quad Serial Peripheral Interface (QuadSPI)

The QuadSPI has the following general features:

- Interface for up to two external quad serial flash memories for code/data storage and code execution
- Supports industry standard: Single, dual and quad mode serial flashes

## 1.4.18 Queue Direct Memory Access Controller (qDMA)

The qDMA controller transfers blocks of data between one source and one or more destinations. The blocks of data transferred can be represented in memory as contiguous or non-contiguous using scatter/gather table(s).

The qDMA supports following general features:

- Supports channel virtualization through enqueueing of DMA jobs to, or dequeuing DMA jobs from, different work queues
- Supports four virtualized blocks for multi core support
- Supports 8 command queues and one status queue per virtualized blocks
- Supports PQ3 legacy direct mode through register interface

If the qDMA is operating in a mixed command queue/legacy mode, legacy mode jobs will be serviced with highest priority as soon as an engine becomes available.

The qDMA is a high performance DMA and can be used for data transfers between DDR to DDR, DDR to PCI Express for outbound transactions and DDR to memory-mapped flash interfaced through IFC.

## 1.4.19 Serial Peripheral Interface (SPI)

The chip supports the synchronous serial bus for communication to an external device.

The module supports the following features:

- Full-duplex, three-wire synchronous transfers
- Buffered transmit operation using the transmit first in first out (TX FIFO) with depth of 16 entries
- Support for 8/16-bit accesses to the PUSH TX FIFO register data field
- Buffered receive operation using the receive FIFO (RX FIFO) with depth of 16 entries
- Asynchronous clocking scheme for register and protocol interfaces

## **1.4.20 Watchdog Timer (WDOG)**

The WDOG module monitors internal system operation and forces a reset in case of failure. It operates on RTC 32 KHz clock. The chip supports five WDOGs, out of which one is dedicated for Trustzone support and other four are for A53 cores (one for each core in the cluster).

# Chapter 2

## Memory Map

### 2.1 Memory map overview

There are several address domains within the chip, including the following:

- Logical, virtual, and physical (real) address spaces within the Arm Architecture core(s)
- Internal configuration, control, and status register (CCSR) address space, which is a special-purpose subset of the internal local address space

#### NOTE

SCFG\_ALTCBAR refers to the higher address bits of the complete CCSR address space, it refers to 01 as the higher address.

- Internal debug control and status register (DCSR) address space, which is another special-purpose set of registers mapped in the internal local address space
- External memory, I/O, and configuration address spaces of the PCI Express links

The MMU in the core handles translation of logical (effective) addresses, into virtual addresses, and ultimately to the physical addresses for the address space. The MMU is described in the core reference manual. When the MMU is configured, the size of the chip's address space is 40-bits providing access to 1 TB; otherwise the address space is 32-bits providing access to 4 GB.

### 2.2 Fixed memory map

This table shows the system memory map for the chip.

Fixed memory map

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

Start Address (Hex)	Module Name	Size	Accessible with x-bit addressing		
			32	36	40
00_0000_0000	Secure Boot ROM	1 MB	Y	Y	Y
00_0010_0000	Extended Boot ROM	15 MB	Y	Y	Y
00_0100_0000	CCSR Register Space	240 MB	Y	Y	Y
00_1000_0000	OCRAM1	64 KB	Y	Y	Y
00_1001_0000	OCRAM2	64 KB	Y	Y	Y
00_1004_0000	Reserved	65408 KB	Y	Y	Y
00_1100_0000	Reserved	16 MB	Y	Y	Y
00_1200_0000	STM	16 MB	Y	Y	Y
00_1300_0000	Reserved	208 MB	Y	Y	Y
00_2000_0000	DCSR	64 MB	Y	Y	Y
00_2400_0000	Reserved	448 MB	Y	Y	Y
00_4000_0000	QuadSPI	512 MB	Y	Y	Y
00_6000_0000	IFC region 1(0-512MB)	512 MB	Y	Y	Y
00_8000_0000	DRAM1 <sup>1</sup> (0-2GB)	2 GB	Y	Y	Y
01_0000_0000	Reserved	0.0625 GB	N	Y	Y
01_0400_0000	Reserved	3.9375 GB	N	Y	Y
02_0000_0000	Reserved	1 GB	N	Y	Y
02_4000_0000	Reserved	7 GB	N	Y	Y
04_0000_0000	Reserved	0.25 GB	N	Y	Y
04_1000_0000	Reserved	0.25 GB	N	Y	Y
04_2000_0000	Reserved	0.25 GB	N	Y	Y
04_3000_0000	Reserved	1.25 GB	N	Y	Y
04_8000_0000	Reserved	2 GB	N	Y	Y
05_0000_0000	QMAN S/W Portal	128 MB	N	Y	Y
05_0800_0000	BMAN S/W Portal	128 MB	N	Y	Y
05_1000_0000	Reserved	4 GB - 256 MB	N	Y	Y
06_0000_0000	Reserved	0.5 GB	N	Y	Y
06_2000_0000	IFC region 2 (512MB-4GB)	3.5 GB	N	Y	Y
07_0000_0000	Reserved	4 GB	N	Y	Y
08_0000_0000	Reserved	2 GB	N	Y	Y
08_8000_0000	DRAM2 <sup>1</sup>	30 GB	N	Y	Y
10_0000_0000	Reserved	64 GB	N	Y	Y
20_0000_0000	Reserved	128 GB	N	N	Y
40_0000_0000	PCI Express 1	32 GB	N	N	Y
48_0000_0000	PCI Express 2	32 GB	N	N	Y

Table continues on the next page...

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

Start Address (Hex)	Module Name	Size	Accessible with x-bit addressing		
			32	36	40
50_0000_0000	PCI Express 3	32 GB	N	N	Y
58_0000_0000	Reserved	160 GB	N	N	Y
80_0000_0000	Reserved	32 GB	N	N	Y
88_0000_0000	DRAM3 <sup>1</sup> (32-512GB)	480 GB	N	N	Y

1. DRAM addresses are remapped, refer section [DDR remapping](#).

## 2.2.1 DDR remapping

The following table remaps the DDR address to the following physical DRAM address. This remapped DRAM addresses are seen by the DDR controller.

### NOTE

The chip's physical memory mapping defines two or more memory mapped segments for DDR memory. To achieve larger than 2 GB of DDR memory on a single chip select and to create a contiguous memory space to interleave DDR memory, a remapping logic is implemented. The remapping logic resides between the chip and DDR. The remapping logic, re-maps the chip-mapped DDR physical address to DDR address and vice versa. The remapping logic is not visible or configurable by software.

**Table 2-2. LS1043A Memory Address Remapping**

Chip address	Segment	Size	Remapped DRAM address
00_8000_0000 - 00_FFFF_FFFF	5-8 (DRAM region 1)	2 GB	00_0000_0000 to 00_7FFF_FFFF
08_0000_0000 - 08_7FFF_FFFF	12 (Reserved)	2 GB	00_0000_0000 to 00_7FFF_FFFF
08_8000_0000 - 0F_FFFF_FFFF	12 (DRAM region 2)	30 GB	00_8000_0000 to 07_FFFF_FFFF
80_0000_0000 - 87_FFFF_FFFF	16 (Reserved)	32 GB	00_0000_0000 to 07_FFFF_FFFF
88_0000_0000 - FF_FFFF_FFFF	16 (DRAM region 3)	480 GB	08_0000_0000 to 7F_FFFF_FFFF

## 2.3 CCSR address map

The following table lists the base address assigned to each block from the base of the 240 MB CCSR space.

### NOTE

Arm® Cortex®-A53 is little endian.

**Table 2-3. CCSR Block Base Address Map**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
100_0000 - 107_FFFF	Reserved	-	-	
108_0000 - 108_FFFF	DDR memory controller	<a href="#">DDR register descriptions</a>	Big-endian (byte swapping required)	
109_0000 - 117_FFFF	Reserved			
118_0000 - 118_FFFF	Arm coherency module (CCI-400)	<a href="#">CCI400 register descriptions</a>	Little-endian (byte swapping not required)	
119_0000 - 13F_FFFF	Reserved	-	-	
140_0000-14F_FFFF	GIC-400	-	Little-endian (byte swapping not required)	See the chapter "Arm modules."
150_0000-150_FFFF	TZASC	<a href="#">Register Descriptions</a>	Little-endian (byte swapping not required)	
151_0000-151_FFFF	Central security unit (CSU)	<a href="#">CSU Memory Map/ Register Definition</a>	Little-endian (byte swapping not required)	
152_0000-152_FFFF	Platform control (Miscellaneous system control module (MSCM))	<a href="#">Miscellaneous System Control Module (MSCM)</a>	Big-endian (byte swapping required)	
153_0000-153_FFFF	IFC	<a href="#">IFC memory map/ register definition</a>	Big-endian (byte swapping required)	
154_0000-154_FFFF	Reserved	-	-	
155_0000-155_FFFF	Quad serial peripheral interface (QuaSPI)	<a href="#">Memory Map and Register Definition</a>	Big-endian (byte swapping required)	
156_0000-156_FFFF	Enhanced secured digital host controller (eSDHC)	<a href="#">eSDHC register descriptions</a>	Big-endian (byte swapping required)	
157_0000-157_FFFF	Supplemental configuration unit (SCFG)	<a href="#">SCFG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
158_0000-160_FFFF	Reserved	-	-	

*Table continues on the next page...*

**Table 2-3. CCSR Block Base Address Map (continued)**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
161_0000-161_FFFF	Pre-Boot loader (PBL)	Reserved address space used as internal PBL commands	-	
162_0000-16F_FFFF	Reserved	-	-	
170_0000-17F_FFFF	SEC <sup>1</sup>	-	Big-endian (byte swapping required)	See the Security reference manual.
180_0000-187_FFFF	Reserved	-	-	-
188_0000-188_FFFF <sup>1</sup>	Queue manager (QMan)			See the DPAA reference manual
189_0000-189_FFFF	Buffer manager (BMan) <sup>1</sup>			See the DPAA reference manual
18A_0000-19F_FFFF	Reserved	-	-	-
1A0_0000-1AF_FFFF	Frame manager (FMan) <sup>1</sup>	-	-	See the DPAA reference manual
1B0_0000-1E7_FFFF	Reserved	-	-	-
1E8_0000-1E8_FFFF	Security fuse processor (SFP)	-	Big-endian (byte swapping required)	See QorIQ Trust Architecture 2.1 User Guide
1E9_0000-1E9_FFFF	Security Monitor	-	Big-endian (byte swapping required)	See the QorIQ Trust Architecture 2.1 User Guide
1EA_0000-1EA_FFF	SerDes control	SerDes register descriptions	Big-endian (byte swapping required)	
1EB_0000-1ED_FFF	Reserved	-		
1EE_0000-1EE_0FF	Device configuration and pin control (DCFG)	Device Configuration/Pin Control Memory Map	Big-endian (byte swapping required)	
1EE_1000-1EE_1FF	Clocking	Clocking Memory Map	Big-endian (byte swapping required)	
1EE_2000-1EE_2FF	Run control/power management (RCPM)	RCPM Memory Map/Register Definition	Big-endian (byte swapping required)	
1EE_3000-1EF_FFF	Reserved	-		
1F0_0000-1F0_FFFF	Thermal monitoring unit (TMU)	TMU register descriptions	Big-endian (byte swapping required)	
1F1_0000-20F_FFFF	Reserved	-		
210_0000-210_FFFF	Serial peripheral interface (SPI)	Memory Map/Register Definition	Big-endian (byte swapping required)	
211_0000-217_FFFF	Reserved	-		
218_0000-218_FFFF	I2C controller 1	Memory map and register definition	Byte accessible	
219_0000-219_FFFF	I2C controller 2	Memory map and register definition	Byte accessible	

Table continues on the next page...

**Table 2-3. CCSR Block Base Address Map (continued)**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
21A_0000-21A_FFFF	I2C controller 3	<a href="#">Memory map and register definition</a>	Byte accessible	
21B_0000-21B_FFFF	I2C controller 4	<a href="#">Memory map and register definition</a>	Byte accessible	
21C_0000-21C_FFF	DUART1	<a href="#">DUART register descriptions</a>	Byte accessible	
21D_0000-21D_FFF	DUART2	<a href="#">DUART register descriptions</a>	Byte accessible	
21E_0000-22F_FFFF	Reserved	-	-	-
230_0000-230_FFFF	General purpose I/O 1 (GPIO1)	<a href="#">GPIO register descriptions</a>	Big-endian (byte swapping required)	
231_0000-231_FFFF	General purpose I/O 2 (GPIO2)	<a href="#">GPIO register descriptions</a>	Big-endian (byte swapping required)	
232_0000-232_FFFF	General purpose I/O 3 (GPIO3)	<a href="#">GPIO register descriptions</a>	Big-endian (byte swapping required)	
233_0000-233_FFFF	General purpose I/O 4 (GPIO4)	<a href="#">GPIO register descriptions</a>	Big-endian (byte swapping required)	
234_0000-23F_FFFF	Reserved	-	-	-
240_0000-27F_FFFF	QUICC engine		Big-endian (byte swapping required)	See the QUICC Engine Block Reference Manual with Protocol Interworking.
280_0000-294_FFFF	Reserved	-	-	-
295_0000-295_FFFF	Low Power universal asynchronous receiver/transmitter 1 (LPUART1)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	This is 32-b mode register accessible IP while DUART is byte accessible
296_0000-296_FFFF	Low Power universal asynchronous receiver/transmitter 2 (LPUART2)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	
297_0000-297_FFFF	Low Power universal asynchronous receiver/transmitter 3 (LPUART3)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	
298_0000-298_FFFF	Low Power universal asynchronous receiver/transmitter 4 (LPUART4)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	
299_0000-299_FFFF	Low Power universal asynchronous receiver/transmitter 5 (LPUART5)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	
29A_0000-29A_FFFF	Low Power universal asynchronous receiver/transmitter 6 (LPUART6)	<a href="#">Register definition</a>	Big-endian (byte swapping required)	
29B_0000-29C_FFFF	Reserved	-	-	-
29D_0000-29D_FFF	FlexTimer module 1 (FTM1)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	

Table continues on the next page...

**Table 2-3. CCSR Block Base Address Map (continued)**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
29E_0000-29E_FFFF	FlexTimer module 2 (FTM2)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
29F_0000-29F_FFFF	FlexTimer module 3 (FTM3)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A0_0000-2A0_FFFF	FlexTimer module 4 (FTM4)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A1_0000-2A1_FFFF	FlexTimer module 5 (FTM5)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A2_0000-2A2_FFFF	FlexTimer module 6 (FTM6)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A3_0000-2A3_FFFF	FlexTimer module 7 (FTM7)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A4_0000-2A4_FFFF	FlexTimer module 8 (FTM8)	<a href="#">Memory map and register definition</a>	Big-endian (byte swapping required)	
2A5_0000-2A6_FFFF	Reserved	-	-	-
2A7_0000-2A7_FFFF	Watchdog timer 3 (WDOG3)	<a href="#">WDOG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
2A8_0000-2A8_FFFF	Watchdog timer 4 (WDOG4)	<a href="#">WDOG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
2A9_0000-2A9_FFFF	Watchdog timer 5 (WDOG5)	<a href="#">WDOG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
2AA_0000-2AC_FFF	Reserved	-		
2AD_0000-2AD_FFF	Watchdog timer 1 (WDOG1)	<a href="#">WDOG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
2AE_0000-2AE_FFF	Watchdog timer 2 (WDOG2)	<a href="#">WDOG Memory Map/ Register Definition</a>	Big-endian (byte swapping required)	
2AF_0000-2AF_FFF	Reserved	-	-	-
2B0_0000-2B0_FFFF	SYS_COUNTER (Secure)	<a href="#">Secure system counter memory map/register definition</a>	Little-endian (byte swapping not required)	
2B1_0000-2B1_FFFF	SYS_COUNTER (Non-secure)	<a href="#">Non-Secure system counter memory map/ register definition</a>	Little-endian (byte swapping not required)	
2B2_0000-2BF_FFFF	Reserved	-	-	-
2C0_0000-2C0_FFF	Enhanced direct memory access (eDMA)	<a href="#">Memory map/register definition</a>	Big-endian (byte swapping required)	32-bit address space size
2C1_0000-2C1_FFF	Direct memory access multiplexer 1 (DMAMUX1)	<a href="#">Memory map/register definition</a>	Big-endian (byte swapping required)	
2C2_0000-2C2_FFF	Direct memory access multiplexer 2 (DMAMUX2)	<a href="#">Memory map/register definition</a>	Big-endian (byte swapping required)	
2C3_0000-2C9_FFF	Reserved	-	-	-

Table continues on the next page...

**Table 2-3. CCSR Block Base Address Map (continued)**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
2CA_0000-2CA_FFF F	Interconnect fabric	<a href="#">IF Memory Map/ Register Definition</a>	Little-endian (byte swapping not required)	
2CB_0000-2EF_FFF F	Reserved	-	-	-
2F0_0000-2FF_FFFF	USB3.0 controller 1	<a href="#">USB Memory Map/ Register Definition</a>	Little-endian (byte swapping not required)	Supports 64-bit addressing
300_0000-30F_FFFF	USB3.0 controller 2	<a href="#">USB Memory Map/ Register Definition</a>	Little-endian (byte swapping not required)	Supports 64-bit addressing
310_0000-31F_FFFF	USB3.0 controller 3	<a href="#">USB Memory Map/ Register Definition</a>	Little-endian (byte swapping not required)	Supports 64-bit addressing
320_0000-320_FFFF	SATA	<a href="#">SATA AHCI register descriptions</a>	Little-endian (byte swapping not required)	
321_0000-33F_FFFF	Reserved	-	-	-
340_0000-340_FFFF	PCI Express controller 1	<a href="#">PEX register descriptions</a>	Little-endian (byte swapping not required)	
341_0000-341_FFFF	PCI Express 1 LUT	<a href="#">PEX_LUT register descriptions</a>	Big-endian (byte swapping required)	
350_0000-350_FFFF	PCI Express controller 2	<a href="#">PEX register descriptions</a>	Little-endian (byte swapping not required)	
351_0000-351_FFFF	PCI Express 2 LUT	<a href="#">PEX_LUT register descriptions</a>	Big-endian (byte swapping required)	
352_0000-35F_FFFF	Reserved	-	-	
360_0000-360_FFFF	PCI Express controller 3	<a href="#">PEX register descriptions</a>	Little-endian (byte swapping not required)	
361_0000-361_FFFF	PCI Express 3 LUT	<a href="#">PEX_LUT register descriptions</a>	Big-endian (byte swapping required)	
362_0000-837_FFFF	Reserved	-	-	
838_0000-83F_FFFF	Queue direct memory access controller (qDMA)	<a href="#">qDMA register descriptions</a>	Big-endian (byte swapping required)	
840_0000-84E_FFFF	Reserved	-	-	
84F_0000-84F_FFFF	USB3 #1 PHY (debug i/f)	<a href="#">USB PHY Memory Map/Register Definition</a>	Big-endian (byte swapping required)	
850_0000-850_FFFF	USB3 #2 PHY (debug i/f)	<a href="#">USB PHY Memory Map/Register Definition</a>	Big-endian (byte swapping required)	
851_0000-851_FFFF	USB3 #3 PHY (debug i/f)	<a href="#">USB PHY Memory Map/Register Definition</a>	Big-endian (byte swapping required)	
852_0000-8FF_FFFF	Reserved	-	-	-

*Table continues on the next page...*

**Table 2-3. CCSR Block Base Address Map (continued)**

Block Base Address (Hex)	Block	Sections	CCSR configuration bus endianness	Comments
900_0000-9FF_FFFF	SMMU	-	Little-endian (byte swapping not required)	See the chapter "Arm modules."
A00_0000-FFF_FFFF	Reserved	-	-	-

1. The DPAA components should be in big-endian mode. The DPAA (FMan, QMan, BMan, and Security modules) data structures can be in the following locations:

- CCSR registers
- Portals
- Frame descriptors such as data structures shared between software/hardware (in DDR):
  - Arm A53, little-endian mode: The DPAA software should perform endianness-related byte-swap (for write access, little endian should be swapped to big endian; for read access, big endian should be swapped to little endian) for accessing the DPAA components.
  - Arm A53, big-endian mode: The DPAA software will not perform endianness-related byte-swap for accessing the DPAA components.

## 2.4 Source ID Assignments

This table shows the source ID assignments for LS1043A.

**Table 2-4. Source ID Assignments**

Transaction Source	Source ID
PCI Express 1	8'h00
PCI Express 2	8'h01
PCI Express 3	8'h02
BMan	8'h18
SMMU	8'h1C
SEC	8'h21
QMan	8'h3C
USB 3.0 controller 1	8'h40
USB 3.0 controller 2	8'h41
USB 3.0 controller 3	8'h42
eSDHC	8'h44
QE	8'h46
PBL	8'h48
Reserved	8'h49
Reserved	8'h4A
Reserved	8'h4C
Reserved	8'h4D
SATA	8'h60

*Table continues on the next page...*

**Table 2-4. Source ID Assignments (continued)**

Transaction Source	Source ID
qDMA	8'h70
eDMA	8'h71
FMan	8'hC0 - 8'hCF
Core-Cluster - Core 0	8'hD0
Core-Cluster - Core 1	8'hD1
Core-Cluster - Core 2	8'hD2
Core-Cluster - Core 3	8'hD3

# **Chapter 3**

## **Signal Descriptions**

### **3.1 Signals introduction**

This chapter describes the external signals and is organized into the following sections:

- Overview of signals and cross-references for signals that serve multiple functions
- List of reset configuration signals
- Signal multiplexing details
- List of output signal states at reset

### **3.2 Signals overview**

The signals are grouped as follows:

- DDR memory controller interface signals
- Integrated Flash controller interface signals
- DUART/LPUART interface signals
- I<sup>2</sup>C interface signals
- Enhanced SDHC interface signals
- Serial peripheral interface (SPI) signals
- IEEE 1588 timestamp signals
- Ethernet management interface signals
- Ethernet controller RGMII interface signals
- QE-TDM/HDLC interface signals
- General-purpose input/output, security monitor, system control, power management, and debug signals
- Clock and JTAG signals
- Power-on-reset configuration signals
- QuadSPI interface signals
- FlexTimer module interface signals

## Signals overview

Note that individual chapters of this document provide details for each signal, describing each signal's behavior when the signal is asserted or negated and when the signal is an input or an output.

The following tables provides a summary of the signals grouped by function. This table details the signal name, interface, alternate functions, and whether the signal is an input, output, or bidirectional. The direction of the multiplexed signals applies for the primary signal function listed in the left-most column of the table for that row (and does not apply for the state of the reset configuration signals). Finally, the tables provide a pointer to the table where the signal function is described.

**Table 3-1. LS1043 Signal Reference by Functional Block**

Name	Description	Alternate Function(s)	Pin type
<b>DDR SDRAM Memory Interface 1</b> (See <a href="#">DDR Signals Overview</a> for more details.)			
D1_MA00	Address	-	O
D1_MA01	Address	-	O
D1_MA02	Address	-	O
D1_MA03	Address	-	O
D1_MA04	Address	-	O
D1_MA05	Address	-	O
D1_MA06	Address	-	O
D1_MA07	Address	-	O
D1_MA08	Address	-	O
D1_MA09	Address	-	O
D1_MA10	Address	-	O
D1_MA11	Address	-	O
D1_MA12	Address	-	O
D1_MA13	Address	-	O
D1_MA14	Address	-	O
D1_MA15	Address	-	O
D1_MAPAR_ERR_B	Address Parity Error		I
D1_MAPAR_OUT	Address Parity Out	-	O
D1_MBA0	Bank Select	-	O
D1_MBA1	Bank Select	-	O
D1_MBA2	Bank Select	-	O
D1_MCAS_B	Column Address Strobe	-	O
D1_MCK0	Clock	-	O
D1_MCK0_B	Clock Complement	-	O
D1_MCK1	Clock	-	O
D1_MCK1_B	Clock Complement	-	O
D1_MCKE0	Clock Enable	-	O

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
D1_MCKE1	Clock Enable	-	O
D1_MCS0_B	Chip Select	-	O
D1_MCS1_B	Chip Select	-	O
D1_MCS2_B	Chip Select	-	O
D1_MCS3_B	Chip Select	-	O
D1_MDIC0	Driver Impedance Calibration	-	IO
D1_MDIC1	Driver Impedance Calibration	-	IO
D1_MDM0	Data Mask	-	O
D1_MDM1	Data Mask	-	O
D1_MDM2	Data Mask	-	O
D1_MDM3	Data Mask	-	O
D1_MDM8	Data Mask	-	O
D1_MDQ00	Data	-	IO
D1_MDQ01	Data	-	IO
D1_MDQ02	Data	-	IO
D1_MDQ03	Data	-	IO
D1_MDQ04	Data	-	IO
D1_MDQ05	Data	-	IO
D1_MDQ06	Data	-	IO
D1_MDQ07	Data	-	IO
D1_MDQ08	Data	-	IO
D1_MDQ09	Data	-	IO
D1_MDQ10	Data	-	IO
D1_MDQ11	Data	-	IO
D1_MDQ12	Data	-	IO
D1_MDQ13	Data	-	IO
D1_MDQ14	Data	-	IO
D1_MDQ15	Data	-	IO
D1_MDQ16	Data	-	IO
D1_MDQ17	Data	-	IO
D1_MDQ18	Data	-	IO
D1_MDQ19	Data	-	IO
D1_MDQ20	Data	-	IO
D1_MDQ21	Data	-	IO
D1_MDQ22	Data	-	IO
D1_MDQ23	Data	-	IO
D1_MDQ24	Data	-	IO
D1_MDQ25	Data	-	IO
D1_MDQ26	Data	-	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
D1_MDQ27	Data	-	IO
D1_MDQ28	Data	-	IO
D1_MDQ29	Data	-	IO
D1_MDQ30	Data	-	IO
D1_MDQ31	Data	-	IO
D1_MDQS0	Data Strobe	-	IO
D1_MDQS0_B	Data Strobe	-	IO
D1_MDQS1	Data Strobe	-	IO
D1_MDQS1_B	Data Strobe	-	IO
D1_MDQS2	Data Strobe	-	IO
D1_MDQS2_B	Data Strobe	-	IO
D1_MDQS3	Data Strobe	-	IO
D1_MDQS3_B	Data Strobe	-	IO
D1_MDQS8	Data Strobe	-	IO
D1_MDQS8_B	Data Strobe	-	IO
D1_MECC0	Error Correcting Code	-	IO
D1_MECC1	Error Correcting Code	-	IO
D1_MECC2	Error Correcting Code	-	IO
D1_MECC3	Error Correcting Code	-	IO
D1_MODT0	On Die Termination	-	O
D1_MODT1	On Die Termination	-	O
D1_MRAS_B	Row Address Strobe	-	O
D1_MWE_B	Write Enable	-	O
<b>Integrated Flash Controller (See <a href="#">External signal descriptions</a> for more details.)</b>			
IFC_A16	IFC Address	QSPI_A_CS0	O
IFC_A17	IFC Address	QSPI_A_CS1	O
IFC_A18	IFC Address	QSPI_A_SCK	O
IFC_A19	IFC Address	QSPI_B_CS0	O
IFC_A20	IFC Address	QSPI_B_CS1	O
IFC_A21	IFC Address	QSPI_B_SCK cfg_dram_type	O
IFC_A22 / IFC_WP1_B	IFC Address	QSPI_A_DATA0	O
IFC_A23 / IFC_WP2_B	IFC Address	QSPI_A_DATA1	O
IFC_A24 / IFC_WP3_B	IFC Address	QSPI_A_DATA2	O
IFC_A25 / IFC_CS4_B / IFC_RB2_B	IFC Address	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	O
IFC_A26 / IFC_CS5_B / IFC_RB3_B	IFC Address	GPIO2_26 FTM5_CH1	O
IFC_A27 / IFC_CS6_B	IFC Address	GPIO2_27	O

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		FTM5_EXTCLK	
<b>IFC_AD00</b>	IFC Address / Data	cfg_gpinput0	IO
<b>IFC_AD01</b>	IFC Address / Data	cfg_gpinput1	IO
<b>IFC_AD02</b>	IFC Address / Data	cfg_gpinput2	IO
<b>IFC_AD03</b>	IFC Address / Data	cfg_gpinput3	IO
<b>IFC_AD04</b>	IFC Address / Data	cfg_gpinput4	IO
<b>IFC_AD05</b>	IFC Address / Data	cfg_gpinput5	IO
<b>IFC_AD06</b>	IFC Address / Data	cfg_gpinput6	IO
<b>IFC_AD07</b>	IFC Address / Data	cfg_gpinput7	IO
<b>IFC_AD08</b>	IFC Address / Data	cfg_rcw_src0	IO
<b>IFC_AD09</b>	IFC Address / Data	cfg_rcw_src1	IO
<b>IFC_AD10</b>	IFC Address / Data	cfg_rcw_src2	IO
<b>IFC_AD11</b>	IFC Address / Data	cfg_rcw_src3	IO
<b>IFC_AD12</b>	IFC Address / Data	cfg_rcw_src4	IO
<b>IFC_AD13</b>	IFC Address / Data	cfg_rcw_src5	IO
<b>IFC_AD14</b>	IFC Address / Data	cfg_rcw_src6	IO
<b>IFC_AD15</b>	IFC Address / Data	cfg_rcw_src7	IO
<b>IFC_AVD</b>	IFC Address Valid		O
<b>IFC_BCTL</b>	IFC Buffer control		O
<b>IFC_CLE</b>	IFC Command Latch Enable / Write Enable	cfg_rcw_src8	O
<b>IFC_CLK0</b>	IFC Clock	-	O
<b>IFC_CLK1</b>	IFC Clock	-	O
<b>IFC_CS0_B</b>	IFC Chip Select	-	O
<b>IFC_CS1_B</b>	IFC Chip Select	GPIO2_10 FTM7_CH0	O
<b>IFC_CS2_B</b>	IFC Chip Select	GPIO2_11 FTM7_CH1	O
<b>IFC_CS3_B</b>	IFC Chip Select	GPIO2_12 QSPI_B_DATA3 FTM7_EXTCLK	O
<b>IFC_CS4_B / IFC_A25 / IFC_RB2_B</b>	IFC Chip Select	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	O
<b>IFC_CS5_B / IFC_A26 / IFC_RB3_B</b>	IFC Chip Select	GPIO2_26 FTM5_CH1	O
<b>IFC_CS6_B / IFC_A27</b>	IFC Chip Select	GPIO2_27 FTM5_EXTCLK	O
<b>IFC_NDDDR_CLK</b>	IFC NAND DDR Clock	-	O
<b>IFC_NDDQS</b>	IFC DQS Strobe	-	IO
<b>IFC_OE_B</b>	IFC Output Enable	cfg_eng_use1	O
<b>IFC_PAR0</b>	IFC Address & Data Parity	GPIO2_13	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		QSPI_B_DATA0 FTM6_CH0	
<b>IFC_PAR1</b>	IFC Address & Data Parity	GPIO2_14 QSPI_B_DATA1 FTM6_CH1	IO
<b>IFC_PERR_B</b>	IFC Parity Error	GPIO2_15 QSPI_B_DATA2 FTM6_EXTCLK	I
<b>IFC_RB0_B</b>	IFC Ready / Busy CS0		I
<b>IFC_RB1_B</b>	IFC Ready / Busy CS1		I
<b>IFC_RB2_B / IFC_A25 / IFC_CS4_B</b>	IFC Ready/Busy CS 2	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	I
<b>IFC_RB3_B / IFC_A26 / IFC_CS5_B</b>	IFC Ready/Busy CS 3	GPIO2_26 FTM5_CH1	I
<b>IFC_TE</b>	IFC External Transceiver Enable	cfg_ifc_te	O
<b>IFC_WE0_B</b>	IFC Write Enable	cfg_eng_use0	O
<b>IFC_WP0_B</b>	IFC Write Protect	cfg_eng_use2	O
<b>IFC_WP1_B / IFC_A22</b>	IFC Write Protect	QSPI_A_DATA0	O
<b>IFC_WP2_B / IFC_A23</b>	IFC Write Protect	QSPI_A_DATA1	O
<b>IFC_WP3_B / IFC_A24</b>	IFC Write Protect	QSPI_A_DATA2	O
<b>DUART (See <a href="#">DUART external signal descriptions</a> for more details.)</b>			
<b>UART1_CTS_B / UART3_SIN</b>	Clear To Send	GPIO1_21 FTM4_CH4 LPUART2_SIN	I
<b>UART1_RTS_B / UART3_SOUT</b>	Ready to Send	GPIO1_19 LPUART2_SOUT FTM4_CH2	O
<b>UART1_SIN</b>	Receive Data	GPIO1_17	I
<b>UART1_SOUT</b>	Transmit Data	GPIO1_15	O
<b>UART2_CTS_B / UART4_SIN</b>	Clear To Send	GPIO1_22 FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	I
<b>UART2_RTS_B / UART4_SOUT</b>	Ready to Send	GPIO1_20 LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	O
<b>UART2_SIN</b>	Receive Data	GPIO1_18 FTM4_CH1 LPUART1_SIN	I
<b>UART2_SOUT</b>	Transmit Data	GPIO1_16 LPUART1_SOUT FTM4_CH0	O

Table continues on the next page...

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
UART3_SIN / <b>UART1_CTS_B</b>	Receive Data	GPIO1_21 FTM4_CH4 LPUART2_SIN	I
UART3_SOUT / <b>UART1_RTS_B</b>	Transmit Data	GPIO1_19 LPUART2_SOUT FTM4_CH2	O
UART4_SIN / <b>UART2_CTS_B</b>	Receive Data	GPIO1_22 FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	I
UART4_SOUT / <b>UART2_RTS_B</b>	Transmit Data	GPIO1_20 LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	O
<b>SPI Interface</b> (See <a href="#">LS1043A SPI signals</a> for more details.)			
<b>SPI_CLK</b>	SPI Clock		O
<b>SPI_CS0_B</b>	SPI Chip Select	GPIO2_00 SDHC_DAT4 SDHC_VS	O
<b>SPI_CS1_B</b>	SPI Chip Select	GPIO2_01 SDHC_DAT5 SDHC_CMD_DIR	O
<b>SPI_CS2_B</b>	SPI Chip Select	GPIO2_02 SDHC_DAT6 SDHC_DAT0_DIR	O
<b>SPI_CS3_B</b>	SPI Chip Select	GPIO2_03 SDHC_DAT7 SDHC_DAT123_DIR	O
<b>SPI_MISO</b>	Master In Slave Out	SDHC_CLK_SYNC_IN	I
<b>SPI_MOSI</b>	Master Out Slave In	SDHC_CLK_SYNC_OUT	O
<b>eSDHC</b> (See <a href="#">eSDHC signal descriptions</a> for more details.)			
<b>SDHC_CD_B</b>	Command	IIC2_SCL GPIO4_02 FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	I
<b>SDHC_CLK</b>	Host to Card Clock	GPIO2_09 LPUART3_CTS_B LPUART6_SIN FTM4_QD_PHB	O
SDHC_CLK_SYNC_IN	IN	<b>SPI_MISO</b>	I
SDHC_CLK_SYNC_OUT	OUT	<b>SPI_MOSI</b>	O
<b>SDHC_CMD</b>	Command/Response	GPIO2_04	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		LPUART3_SOUT FTM4_CH6	
SDHC_CMD_DIR	DIR	SPI_CS1_B GPIO2_01 SDHC_DAT5	O
SDHC_DAT0	Data	GPIO2_05 FTM4_CH7 LPUART3_SIN	IO
SDHC_DAT0_DIR	DIR	SPI_CS2_B GPIO2_02 SDHC_DAT6	O
SDHC_DAT1	Data	GPIO2_06 LPUART5_SOUT FTM4_FAULT LPUART2_RTS_B	IO
SDHC_DAT123_DIR	DIR	SPI_CS3_B GPIO2_03 SDHC_DAT7	O
SDHC_DAT2	Data	GPIO2_07 LPUART2_CTS_B LPUART5_SIN FTM4_EXTCLK	IO
SDHC_DAT3	Data	GPIO2_08 LPUART6_SOUT FTM4_QD_PHA LPUART3_RTS_B	IO
SDHC_DAT4	Data	SPI_CS0_B GPIO2_00 SDHC_VS	IO
SDHC_DAT5	Data	SPI_CS1_B GPIO2_01 SDHC_CMD_DIR	IO
SDHC_DAT6	Data	SPI_CS2_B GPIO2_02 SDHC_DAT0_DIR	IO
SDHC_DAT7	Data	SPI_CS3_B GPIO2_03 SDHC_DAT123_DIR	IO
SDHC_VS	VS	SPI_CS0_B GPIO2_00 SDHC_DAT4	O
SDHC_WP	Write Protect	IIC2_SDA GPIO4_03 FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	I
<b>Programmable Interrupt Controller</b>			

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
EVT9_B	Interrupt Output		O
IRQ00	External Interrupt		I
IRQ01	External Interrupt		I
IRQ02	External Interrupt		I
IRQ03	External Interrupt	GPIO1_23 FTM3_CH7 TDMB_TSYNC UC3_RTSB_TXEN	I
IRQ04	External Interrupt	GPIO1_24 FTM3_CH0 TDMA_RXD UC1_RXD7 TDMA_RXD	I
IRQ05	External Interrupt	GPIO1_25 FTM3_CH1 TDMA_RSYNC UC1_CTSB_RXDV	I
IRQ06	External Interrupt	GPIO1_26 FTM3_CH2 TDMA_RXD_EXC TDMA_RXD UC1_RXD7	I
IRQ07	External Interrupt	GPIO1_27 FTM3_CH3 TDMA_TSYNC UC1_RTSB_TXEN	I
IRQ08	External Interrupt	GPIO1_28 FTM3_CH4 TDMB_RXD UC3_RXD7 TDMB_RXD	I
IRQ09	External Interrupt	GPIO1_29 FTM3_CH5 TDMB_RSYNC UC3_CTSB_RXDV	I
IRQ10	External Interrupt	GPIO1_30 FTM3_CH6 TDMB_RXD_EXC TDMB_RXD UC3_RXD7	I
IRQ11	External Interrupt	GPIO1_31	I
<b>Battery Backed Trust</b>			
TA_BB_TMP_DETECT_B	Battery Backed Tamper Detect	-	I
<b>Trust</b>			
TA_TMP_DETECT_B	Tamper Detect		I
<b>System Control</b> (See <a href="#">External signal descriptions</a> for more details.)			

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>HRESET_B</b>	Hard Reset	-	IO
<b>PORESET_B</b>	Power On Reset	-	I
<b>RESET_REQ_B</b>	Reset Request (POR or Hard)		O
<b>Power Management</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>ASLEEP</b>	Asleep	GPIO1_13	O
<b>SYCLK</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>SYSCLK</b>	System Clock	-	I
<b>DDR Clocking</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>DDRCLK</b>	DDR Controller Clock	-	I
<b>RTC</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>RTC</b>	Real Time Clock	GPIO1_14	I
<b>Debug</b>			
<b>CKSTP_OUT_B</b>	This signal is reserved for internal use. Refer the chip specific design checklist for more information.	-	O
<b>CLK_OUT</b>	Clock Out	-	O
<b>EVT0_B</b>	Event 0	-	IO
<b>EVT1_B</b>	Event 1	-	IO
<b>EVT2_B</b>	Event 2	-	IO
<b>EVT3_B</b>	Event 3	-	IO
<b>EVT4_B</b>	Event 4	-	IO
<b>EVT5_B</b>	Event 5	<b>IIC3_SCL</b> GPIO4_10 USB2_DRVVBUS BRGO4 FTM8_CH0 CLK11	IO
<b>EVT6_B</b>	Event 6	<b>IIC3_SDA</b> GPIO4_11 USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
<b>EVT7_B</b>	Event 7	<b>IIC4_SCL</b> GPIO4_12 USB3_DRVVBUS TDMA_RQ FTM3_FAULT UC1_CDB_RXER	IO
<b>EVT8_B</b>	Event 8	<b>IIC4_SDA</b> GPIO4_13 USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
<b>DFT</b>			

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
JTAG_BSR_VSEL	An IEEE 1149.1 JTAG compliance enable pin. 0: Normal operation. 1: To be compliant to the 1149.1 specification for boundary scan functions. The JTAG compliant state is documented in the BSDL.  <b>NOTE:</b> When this pin is tied high, the USB PHY will not come out or reset in normal functional mode.	-	I
SCAN_MODE_B	Reserved	-	I
TBSCAN_EN_B	An IEEE 1149.1 JTAG compliance enable pin. 0:To be compliant to the 1149.1 specification for boundary scan functions. The JTAG compliant state is documented in the BSDL. 1: JTAG connects to DAP controller for the Arm core debug.	-	I
TEST_SEL_B	Reserved	-	I
<b>JTAG</b>			
TCK	Test Clock	-	I
TDI	Test Data In	-	I
TDO	Test Data Out	-	O
TMS	Test Mode Select	-	I
TRST_B	Test Reset	-	I
<b>Analog Signals</b>			
D1_MVREF	SSTL Reference Voltage	-	IO
D1_TPA	DDR Controller 1 Test Point Analog	-	IO
FA_ANALOG_G_V	Reserved	-	IO
FA_ANALOG_PIN	Reserved	-	IO
TD1_ANODE	Thermal diode anode	-	IO
TD1_CATHODE	Thermal diode cathode	-	IO
TH_TPA	Thermal Test Point Analog	-	-
<b>SerDes</b> (See <a href="#">External Signals Description</a> for more details.)			
SD1_IMP_CAL_RX	SerDes Receive Impedance Calibration	-	I
SD1_IMP_CAL_TX	SerDes Transmit Impedance Calibration	-	I
SD1_PLL1_TPA	SerDes PLL 1 Test Point Analog	-	O
SD1_PLL1_TPD	SerDes Test Point Digital	-	O
SD1_PLL2_TPA	SerDes PLL 2 Test Point Analog	-	O
SD1_PLL2_TPD	SerDes Test Point Digital	-	O
SD1_REF_CLK1_N	SerDes PLL 1 Reference Clock Complement	-	I
SD1_REF_CLK1_P	SerDes PLL 1 Reference Clock	-	I
SD1_REF_CLK2_N	SerDes PLL 2 Reference Clock Complement	-	I
SD1_REF_CLK2_P	SerDes PLL 2 Reference Clock	-	I
SD1_RX0_N	SerDes Receive Data (negative)	-	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>SD1_RX0_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX1_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX1_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX2_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX2_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX3_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX3_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_TX0_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX0_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX1_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX1_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX2_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX2_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX3_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX3_P</b>	SerDes Transmit Data (positive)	-	O
<b>USB3 PHY 1</b> (See <a href="#">External Signals</a> for more details.)			
<b>USB1_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB1_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB1_ID</b>	USB PHY ID Detect	-	I
<b>USB1_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB1_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB1_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB1_TX_M</b>	USB PHY SS Transmit Data (-)	-	O
<b>USB1_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB1_VBUS</b>	USB PHY VBUS	-	I
<b>USB3 PHY 2</b> (See <a href="#">External Signals</a> for more details.)			
<b>USB2_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB2_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB2_ID</b>	USB PHY ID Detect	-	I
<b>USB2_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB2_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB2_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB2_TX_M</b>	USB PHY SS Transmit Data (-)	-	O
<b>USB2_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB2_VBUS</b>	USB PHY VBUS	-	I
<b>USB3 PHY 3</b> (See <a href="#">External Signals</a> for more details.)			
<b>USB3_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB3_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB3_ID</b>	USB PHY ID Detect	-	I

Table continues on the next page...

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>USB3_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB3_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB3_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB3_TX_M</b>	USB PHY SS Transmit Data (-)	-	O
<b>USB3_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB3_VBUS</b>	USB PHY VBUS	-	I
<b>Ethernet Management Interface 1</b> (See DPAA reference manual for more details.)			
<b>EMI1_MDC</b>	Management Data Clock	GPIO3_00	O
<b>EMI1_MDIO</b>	Management Data In/Out	GPIO3_01	IO
<b>Ethernet Management Interface 2</b> (See DPAA reference manual for more details.)			
<b>EMI2_MDC</b>	Management Data Clock	GPIO4_00	O
<b>EMI2_MDIO</b>	Management Data In/Out	GPIO4_01	IO
<b>Ethernet Controller 1</b> (See DPAA reference manual for more details.)			
<b>EC1_GTX_CLK</b>	Transmit Clock Out	GPIO3_07 FTM1_EXTCLK	O
<b>EC1_GTX_CLK125</b>	Reference Clock	GPIO3_08	I
<b>EC1_RXD0</b>	Receive Data	GPIO3_12 FTM1_CH0	I
<b>EC1_RXD1</b>	Receive Data	GPIO3_11 FTM1_CH1	I
<b>EC1_RXD2</b>	Receive Data	GPIO3_10 FTM1_CH6	I
<b>EC1_RXD3</b>	Receive Data	GPIO3_09 FTM1_CH4	I
<b>EC1_RX_CLK</b>	Receive Clock	GPIO3_13 FTM1_QD_PHA	I
<b>EC1_RX_DV</b>	Receive Data Valid	GPIO3_14 FTM1_QD_PHB	I
<b>EC1_TXD0</b>	Transmit Data	GPIO3_05 FTM1_CH2	O
<b>EC1_TXD1</b>	Transmit Data	GPIO3_04 FTM1_CH3	O
<b>EC1_TXD2</b>	Transmit Data	GPIO3_03 FTM1_CH7	O
<b>EC1_TXD3</b>	Transmit Data	GPIO3_02 FTM1_CH5	O
<b>EC1_TX_EN</b>	Transmit Enable	GPIO3_06 FTM1_FAULT	O
<b>Ethernet Controller 2</b> (See DPAA reference manual for more details.)			
<b>EC2_GTX_CLK</b>	Transmit Clock Out	GPIO3_20 FTM2_EXTCLK	O
<b>EC2_GTX_CLK125</b>	Reference Clock	GPIO3_21	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>EC2_RXD0</b>	Receive Data	GPIO3_25 TSEC_1588_TRIG_IN2 FTM2_CH0	I
<b>EC2_RXD1</b>	Receive Data	GPIO3_24 TSEC_1588_PULS_E_OUT1 FTM2_CH1	I
<b>EC2_RXD2</b>	Receive Data	GPIO3_23 FTM2_CH6	I
<b>EC2_RXD3</b>	Receive Data	GPIO3_22 FTM2_CH4	I
<b>EC2_RX_CLK</b>	Receive Clock	GPIO3_26 TSEC_1588_CLK_IN FTM2_QD_PHA	I
<b>EC2_RX_DV</b>	Receive Data Valid	GPIO3_27 TSEC_1588_TRIG_IN1 FTM2_QD_PHB	I
<b>EC2_TXD0</b>	Transmit Data	GPIO3_18 TSEC_1588_PULS_E_OUT2 FTM2_CH2	O
<b>EC2_TXD1</b>	Transmit Data	GPIO3_17 TSEC_1588_CLK_OUT FTM2_CH3	O
<b>EC2_TXD2</b>	Transmit Data	GPIO3_16 TSEC_1588_ALARM_OUT1 FTM2_CH7	O
<b>EC2_TXD3</b>	Transmit Data	GPIO3_15 TSEC_1588_ALARM_OUT2 FTM2_CH5	O
<b>EC2_TX_EN</b>	Transmit Enable	GPIO3_19 FTM2_FAULT	O
<b>I2C</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>IIC1_SCL</b>	Serial Clock (supports PBL)		IO
<b>IIC1_SDA</b>	Serial Data (supports PBL)		IO
<b>IIC2_SCL</b>	Serial Clock	GPIO4_02 SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	IO
<b>IIC2_SDA</b>	Serial Data	GPIO4_03	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	
IIC3_SCL	Serial Clock	GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0 CLK11	IO
IIC3_SDA	Serial Data	GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
IIC4_SCL	Serial Clock	GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ FTM3_FAULT UC1_CDB_RXER	IO
IIC4_SDA	Serial Data	GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
<b>USB</b> (See <a href="#">External Signals</a> for more details.)			
USB2_DRVVBUS	DRV VBus	IIC3_SCL GPIO4_10 EVT5_B BRGO4 FTM8_CH0 CLK11	O
USB2_PWRFAULT	PWR Fault	IIC3_SDA GPIO4_11 EVT6_B BRGO1 FTM8_CH1 CLK12_CLK8	I
USB3_DRVVBUS	DRV Bus	IIC4_SCL GPIO4_12 EVT7_B TDMA_RQ FTM3_FAULT UC1_CDB_RXER	O
USB3_PWRFAULT	PWR Fault	IIC4_SDA GPIO4_13 EVT8_B	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	
<b>USB_DRVVBUS</b>	USB_DRVVBUS	GPIO4_29	O
<b>USB_PWRFAULT</b>	USB_PWRFAULT	GPIO4_30	I
<b>Battery Backed RTC</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>TA_BB_RTC</b>	This signal is reserved for internal information. Refer the chip specific design checklist for more information.	-	I
<b>D SYSCLK</b> (See <a href="#">External signal descriptions</a> for more details.)			
<b>DIFF_SYSCLK</b>	Single Source System Clock Differential (positive)	-	I
<b>DIFF_SYSCLK_B</b>	Single Source System Clock Differential (negative)	-	I
<b>Power-On-Reset Configuration</b> (See <a href="#">Configuration signals sampled at reset</a> for more details.)			
cfg_dram_type	Power-on-Reset Configuration	<b>IFC_A21</b> QSPI_B_SCK	I
cfg_eng_use0	Power-on-Reset Configuration	<b>IFC_WE0_B</b>	I
cfg_eng_use1	Power-on-Reset Configuration	<b>IFC_OE_B</b>	I
cfg_eng_use2	Power-on-Reset Configuration	<b>IFC_WP0_B</b>	I
cfg_gpininput0	Power-on-Reset Configuration	<b>IFC_AD00</b>	I
cfg_gpininput1	Power-on-Reset Configuration	<b>IFC_AD01</b>	I
cfg_gpininput2	Power-on-Reset Configuration	<b>IFC_AD02</b>	I
cfg_gpininput3	Power-on-Reset Configuration	<b>IFC_AD03</b>	I
cfg_gpininput4	Power-on-Reset Configuration	<b>IFC_AD04</b>	I
cfg_gpininput5	Power-on-Reset Configuration	<b>IFC_AD05</b>	I
cfg_gpininput6	Power-on-Reset Configuration	<b>IFC_AD06</b>	I
cfg_gpininput7	Power-on-Reset Configuration	<b>IFC_AD07</b>	I
cfg_ifc_te	Power-on-Reset Configuration	<b>IFC_TE</b>	I
cfg_rcw_src0	Power-on-Reset Configuration	<b>IFC_AD08</b>	I
cfg_rcw_src1	Power-on-Reset Configuration	<b>IFC_AD09</b>	I
cfg_rcw_src2	Power-on-Reset Configuration	<b>IFC_AD10</b>	I
cfg_rcw_src3	Power-on-Reset Configuration	<b>IFC_AD11</b>	I
cfg_rcw_src4	Power-on-Reset Configuration	<b>IFC_AD12</b>	I
cfg_rcw_src5	Power-on-Reset Configuration	<b>IFC_AD13</b>	I
cfg_rcw_src6	Power-on-Reset Configuration	<b>IFC_AD14</b>	I
cfg_rcw_src7	Power-on-Reset Configuration	<b>IFC_AD15</b>	I
cfg_rcw_src8	Power-on-Reset Configuration	<b>IFC_CLE</b>	I
<b>General Purpose Input/Output</b> (See <a href="#">GPIO signal descriptions</a> for more details.)			
GPIO1_13	General Purpose Input/Output	<b>ASLEEP</b>	O
GPIO1_14	General Purpose Input/Output	<b>RTC</b>	IO
GPIO1_15	General Purpose Input/Output	<b>UART1_SOUT</b>	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
GPIO1_16	General Purpose Input/Output	UART2_SOUT LPUART1_SOUT FTM4_CH0	IO
GPIO1_17	General Purpose Input/Output	UART1_SIN	IO
GPIO1_18	General Purpose Input/Output	UART2_SIN FTM4_CH1 LPUART1_SIN	IO
GPIO1_19	General Purpose Input/Output	UART1_RTS_B UART3_SOUT LPUART2_SOUT FTM4_CH2	IO
GPIO1_20	General Purpose Input/Output	UART2_RTS_B UART4_SOUT LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	IO
GPIO1_21	General Purpose Input/Output	UART1_CTS_B UART3_SIN FTM4_CH4 LPUART2_SIN	IO
GPIO1_22	General Purpose Input/Output	UART2_CTS_B UART4_SIN FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	IO
GPIO1_23	General Purpose Input/Output	IRQ03 FTM3_CH7 TDMA_TSYNC UC3_RTSB_TXEN	IO
GPIO1_24	General Purpose Input/Output	IRQ04 FTM3_CH0 TDMA_RXD UC1_RXD7 TDMA_TXD	IO
GPIO1_25	General Purpose Input/Output	IRQ05 FTM3_CH1 TDMA_RSYNC UC1_CTSB_RXDV	IO
GPIO1_26	General Purpose Input/Output	IRQ06 FTM3_CH2 TDMA_RXD_EXC TDMA_TXD UC1_TXD7	IO
GPIO1_27	General Purpose Input/Output	IRQ07 FTM3_CH3 TDMA_TSYNC UC1_RTSB_TXEN	IO
GPIO1_28	General Purpose Input/Output	IRQ08 FTM3_CH4	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMB_RXD UC3_RXD7 TDMB_TXD	
GPIO1_29	General Purpose Input/Output	<b>IRQ09</b> FTM3_CH5 TDMB_RSYNC UC3_CTSB_RXDV	IO
GPIO1_30	General Purpose Input/Output	<b>IRQ10</b> FTM3_CH6 TDMB_RXD_EXC TDMB_TXD UC3_TXD7	IO
GPIO1_31	General Purpose Input/Output	<b>IRQ11</b>	IO
GPIO2_00	General Purpose Input/Output	<b>SPI_CS0_B</b> SDHC_DAT4 SDHC_VS	IO
GPIO2_01	General Purpose Input/Output	<b>SPI_CS1_B</b> SDHC_DAT5 SDHC_CMD_DIR	IO
GPIO2_02	General Purpose Input/Output	<b>SPI_CS2_B</b> SDHC_DAT6 SDHC_DAT0_DIR	IO
GPIO2_03	General Purpose Input/Output	<b>SPI_CS3_B</b> SDHC_DAT7 SDHC_DAT123_DIR	IO
GPIO2_04	General Purpose Input/Output	<b>SDHC_CMD</b> LPUART3_SOUT FTM4_CH6	IO
GPIO2_05	General Purpose Input/Output	<b>SDHC_DAT0</b> FTM4_CH7 LPUART3_SIN	IO
GPIO2_06	General Purpose Input/Output	<b>SDHC_DAT1</b> LPUART5_SOUT FTM4_FAULT LPUART2 RTS_B	IO
GPIO2_07	General Purpose Input/Output	<b>SDHC_DAT2</b> LPUART2_CTS_B LPUART5_SIN FTM4_EXTCLK	IO
GPIO2_08	General Purpose Input/Output	<b>SDHC_DAT3</b> LPUART6_SOUT FTM4_QD_PHA LPUART3 RTS_B	IO
GPIO2_09	General Purpose Input/Output	<b>SDHC_CLK</b> LPUART3_CTS_B LPUART6_SIN FTM4_QD_PHB	IO
GPIO2_10	General Purpose Input/Output	<b>IFC_CS1_B</b>	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		FTM7_CH0	
GPIO2_11	General Purpose Input/Output	IFC_CS2_B FTM7_CH1	IO
GPIO2_12	General Purpose Input/Output	IFC_CS3_B QSPI_B_DATA3 FTM7_EXTCLK	IO
GPIO2_13	General Purpose Input/Output	IFC_PAR0 QSPI_B_DATA0 FTM6_CH0	IO
GPIO2_14	General Purpose Input/Output	IFC_PAR1 QSPI_B_DATA1 FTM6_CH1	IO
GPIO2_15	General Purpose Input/Output	IFC_PERR_B QSPI_B_DATA2 FTM6_EXTCLK	IO
GPIO2_25	General Purpose Input/Output	IFC_A25 QSPI_A_DATA3 FTM5_CH0 IFC_CS4_B IFC_RB2_B	IO
GPIO2_26	General Purpose Input/Output	IFC_A26 FTM5_CH1 IFC_CS5_B IFC_RB3_B	IO
GPIO2_27	General Purpose Input/Output	IFC_A27 FTM5_EXTCLK IFC_CS6_B	IO
GPIO3_00	General Purpose Input/Output	EMI1_MDC	IO
GPIO3_01	General Purpose Input/Output	EMI1_MDIO	IO
GPIO3_02	General Purpose Input/Output	EC1_TXD3 FTM1_CH5	IO
GPIO3_03	General Purpose Input/Output	EC1_TXD2 FTM1_CH7	IO
GPIO3_04	General Purpose Input/Output	EC1_TXD1 FTM1_CH3	IO
GPIO3_05	General Purpose Input/Output	EC1_TXD0 FTM1_CH2	IO
GPIO3_06	General Purpose Input/Output	EC1_TX_EN FTM1_FAULT	IO
GPIO3_07	General Purpose Input/Output	EC1_GTX_CLK FTM1_EXTCLK	IO
GPIO3_08	General Purpose Input/Output	EC1_GTX_CLK12_5	IO
GPIO3_09	General Purpose Input/Output	EC1_RXD3 FTM1_CH4	IO
GPIO3_10	General Purpose Input/Output	EC1_RXD2 FTM1_CH6	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
GPIO3_11	General Purpose Input/Output	<b>EC1_RXD1</b> FTM1_CH1	IO
GPIO3_12	General Purpose Input/Output	<b>EC1_RXD0</b> FTM1_CH0	IO
GPIO3_13	General Purpose Input/Output	<b>EC1_RX_CLK</b> FTM1_QD_PHA	IO
GPIO3_14	General Purpose Input/Output	<b>EC1_RX_DV</b> FTM1_QD_PHB	IO
GPIO3_15	General Purpose Input/Output	<b>EC2_TXD3</b> TSEC_1588_ALAR M_OUT2 FTM2_CH5	IO
GPIO3_16	General Purpose Input/Output	<b>EC2_TXD2</b> TSEC_1588_ALAR M_OUT1 FTM2_CH7	IO
GPIO3_17	General Purpose Input/Output	<b>EC2_TXD1</b> TSEC_1588_CLK_OUT FTM2_CH3	IO
GPIO3_18	General Purpose Input/Output	<b>EC2_TXD0</b> TSEC_1588_PULS E_OUT2 FTM2_CH2	IO
GPIO3_19	General Purpose Input/Output	<b>EC2_TX_EN</b> FTM2_FAULT	IO
GPIO3_20	General Purpose Input/Output	<b>EC2_GTX_CLK</b> FTM2_EXTCLK	IO
GPIO3_21	General Purpose Input/Output	<b>EC2_GTX_CLK12</b> 5	IO
GPIO3_22	General Purpose Input/Output	<b>EC2_RXD3</b> FTM2_CH4	IO
GPIO3_23	General Purpose Input/Output	<b>EC2_RXD2</b> FTM2_CH6	IO
GPIO3_24	General Purpose Input/Output	<b>EC2_RXD1</b> TSEC_1588_PULS E_OUT1 FTM2_CH1	IO
GPIO3_25	General Purpose Input/Output	<b>EC2_RXD0</b> TSEC_1588_TRIG_IN2 FTM2_CH0	IO
GPIO3_26	General Purpose Input/Output	<b>EC2_RX_CLK</b> TSEC_1588_CLK_IN FTM2_QD_PHA	IO
GPIO3_27	General Purpose Input/Output	<b>EC2_RX_DV</b>	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TSEC_1588_TRIG _IN1 FTM2_QD_PHB	
GPIO4_00	General Purpose Input/Output	<b>EMI2_MDC</b>	IO
GPIO4_01	General Purpose Input/Output	<b>EMI2_MDIO</b>	IO
GPIO4_02	General Purpose Input/Output	<b>IIC2_SCL</b> SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	IO
GPIO4_03	General Purpose Input/Output	<b>IIC2_SDA</b> SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	IO
GPIO4_10	General Purpose Input/Output	<b>IIC3_SCL</b> EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0 CLK11	IO
GPIO4_11	General Purpose Input/Output	<b>IIC3_SDA</b> EVT6_B USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
GPIO4_12	General Purpose Input/Output	<b>IIC4_SCL</b> EVT7_B USB3_DRVVBUS TDMA_RQ FTM3_FAULT UC1_CDB_RXER	IO
GPIO4_13	General Purpose Input/Output	<b>IIC4_SDA</b> EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
GPIO4_29	General Purpose Input/Output	<b>USB_DRVVBUS</b>	IO
GPIO4_30	General Purpose Input/Output	<b>USB_PWRFAULT</b>	IO
<b>Frequency Timer Module 1</b> (See <a href="#">LS1043A FlexTimer signals</a> for more details.)			
FTM1_CH0	Channel 0	<b>EC1_RXD0</b> GPIO3_12	IO
FTM1_CH1	Channel 1	<b>EC1_RXD1</b> GPIO3_11	IO
FTM1_CH2	Channel 2	<b>EC1_TXD0</b>	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		GPIO3_05	
FTM1_CH3	Channel 3	<b>EC1_TXD1</b> GPIO3_04	IO
FTM1_CH4	Channel 4	<b>EC1_RXD3</b> GPIO3_09	IO
FTM1_CH5	Channel 5	<b>EC1_TXD3</b> GPIO3_02	IO
FTM1_CH6	Channel 6	<b>EC1_RXD2</b> GPIO3_10	IO
FTM1_CH7	Channel 7	<b>EC1_TXD2</b> GPIO3_03	IO
FTM1_EXTCLK	Ext Clock	<b>EC1_GTX_CLK</b> GPIO3_07	I
FTM1_FAULT	Fault	<b>EC1_TX_EN</b> GPIO3_06	I
FTM1_QD_PHA	Phase A	<b>EC1_RX_CLK</b> GPIO3_13	I
FTM1_QD_PHB	Phase B	<b>EC1_RX_DV</b> GPIO3_14	I
<b>Frequency Timer Module 2</b> (See <a href="#">LS1043A FlexTimer signals</a> for more details.)			
FTM2_CH0	Channel 0	<b>EC2_RXD0</b> GPIO3_25 TSEC_1588_TRIG_IN2	IO
FTM2_CH1	Channel 1	<b>EC2_RXD1</b> GPIO3_24 TSEC_1588_PULS_E_OUT1	IO
FTM2_CH2	Channel 2	<b>EC2_TXD0</b> GPIO3_18 TSEC_1588_PULS_E_OUT2	IO
FTM2_CH3	Channel 3	<b>EC2_TXD1</b> GPIO3_17 TSEC_1588_CLK_OUT	IO
FTM2_CH4	Channel 4	<b>EC2_RXD3</b> GPIO3_22	IO
FTM2_CH5	Channel 5	<b>EC2_TXD3</b> GPIO3_15 TSEC_1588_ALARM_OUT2	IO
FTM2_CH6	Channel 6	<b>EC2_RXD2</b> GPIO3_23	IO
FTM2_CH7	Channel 7	<b>EC2_TXD2</b> GPIO3_16	IO

Table continues on the next page...

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TSEC_1588_ALAR M_OUT1	
FTM2_EXTCLK	Ext Clock	EC2_GTX_CLK GPIO3_20	I
FTM2_FAULT	Fault	EC2_TX_EN GPIO3_19	I
FTM2_QD_PHA	Phase A	EC2_RX_CLK GPIO3_26 TSEC_1588_CLK_IN	I
FTM2_QD_PHB	Phase B	EC2_RX_DV GPIO3_27 TSEC_1588_TRIG_IN1	I
<b>Frequency Timer Module 3</b> (See <a href="#">LS1043A FlexTimer signals</a> for more details.)			
FTM3_CH0	Channel 0	IRQ04 GPIO1_24 TDMA_RXD UC1_RXD7 TDMA_TXD	IO
FTM3_CH1	Channel 1	IRQ05 GPIO1_25 TDMA_RSYNC UC1_CTSB_RXDV	IO
FTM3_CH2	Channel 2	IRQ06 GPIO1_26 TDMA_RXD_EXC TDMA_TXD UC1_RXD7	IO
FTM3_CH3	Channel 3	IRQ07 GPIO1_27 TDMA_TSYNC UC1_RTSB_TXEN	IO
FTM3_CH4	Channel 4	IRQ08 GPIO1_28 TDMB_RXD UC3_RXD7 TDMB_TXD	IO
FTM3_CH5	Channel 5	IRQ09 GPIO1_29 TDMB_RSYNC UC3_CTSB_RXDV	IO
FTM3_CH6	Channel 6	IRQ10 GPIO1_30 TDMB_RXD_EXC TDMB_TXD UC3_RXD7	IO
FTM3_CH7	Channel 7	IRQ03 GPIO1_23	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMB_TSYNC UC3_RTSB_TXEN	
FTM3_EXTCLK	Ext Clock	IIC4_SDA GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ UC3_CDB_RXER	I
FTM3_FAULT	Fault	IIC4_SCL GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ UC1_CDB_RXER	I
FTM3_QD_PHA	Phase A	IIC2_SCL GPIO4_02 SDHC_CD_B CLK9 QE_SI1_STROBE0 BRGO2	I
FTM3_QD_PHB	Phase B	IIC2_SDA GPIO4_03 SDHC_WP CLK10 QE_SI1_STROBE1 BRGO3	I
<b>Frequency Timer Module 4</b> (See <a href="#">LS1043A FlexTimer signals</a> for more details.)			
FTM4_CH0	Channel 0	UART2_SOUT GPIO1_16 LPUART1_SOUT	IO
FTM4_CH1	Channel 1	UART2_SIN GPIO1_18 LPUART1_SIN	IO
FTM4_CH2	Channel 2	UART1 RTS_B GPIO1_19 UART3_SOUT LPUART2_SOUT	IO
FTM4_CH3	Channel 3	UART2 RTS_B GPIO1_20 UART4_SOUT LPUART4_SOUT LPUART1_RTS_B	IO
FTM4_CH4	Channel 4	UART1 CTS_B GPIO1_21 UART3_SIN LPUART2_SIN	IO
FTM4_CH5	Channel 5	UART2 CTS_B GPIO1_22 UART4_SIN LPUART1_CTS_B	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		LPUART4_SIN	
FTM4_CH6	Channel 6	SDHC_CMD GPIO2_04 LPUART3_SOUT	IO
FTM4_CH7	Channel 7	SDHC_DAT0 GPIO2_05 LPUART3_SIN	IO
FTM4_EXTCLK	Ext Clock	SDHC_DAT2 GPIO2_07 LPUART2_CTS_B LPUART5_SIN	I
FTM4_FAULT	Fault	SDHC_DAT1 GPIO2_06 LPUART5_SOUT LPUART2_RTS_B	I
FTM4_QD_PHA	Phase A	SDHC_DAT3 GPIO2_08 LPUART6_SOUT LPUART3_RTS_B	I
FTM4_QD_PHB	Phase B	SDHC_CLK GPIO2_09 LPUART3_CTS_B LPUART6_SIN	I
<b>Frequency Timer Module 5 (See <a href="#">LS1043A FlexTimer signals</a> for more details.)</b>			
FTM5_CH0	Channel 0	IFC_A25 GPIO2_25 QSPI_A_DATA3 IFC_CS4_B IFC_RB2_B	IO
FTM5_CH1	Channel 1	IFC_A26 GPIO2_26 IFC_CS5_B IFC_RB3_B	IO
FTM5_EXTCLK	Ext Clock	IFC_A27 GPIO2_27 IFC_CS6_B	I
<b>Frequency Timer Module 6 (See <a href="#">LS1043A FlexTimer signals</a> for more details.)</b>			
FTM6_CH0	Channel 0	IFC_PAR0 GPIO2_13 QSPI_B_DATA0	IO
FTM6_CH1	Channel 1	IFC_PAR1 GPIO2_14 QSPI_B_DATA1	IO
FTM6_EXTCLK	Ext Clock	IFC_PERR_B GPIO2_15 QSPI_B_DATA2	I
<b>Frequency Timer Module 7 (See <a href="#">LS1043A FlexTimer signals</a> for more details.)</b>			
FTM7_CH0	Channel 0	IFC_CS1_B	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		GPIO2_10	
FTM7_CH1	Channel 1	IFC_CS2_B GPIO2_11	IO
FTM7_EXTCLK	Ext Clock	IFC_CS3_B GPIO2_12 QSPI_B_DATA3	I
<b>Frequency Timer Module 8</b> (See <a href="#">LS1043A FlexTimer signals</a> for more details.)			
FTM8_CH0	Channel 0	IIC3_SCL GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 CLK11	IO
FTM8_CH1	Channel 1	IIC3_SDA GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1 CLK12_CLK8	IO
<b>LPUART</b> (See <a href="#">Chip LPUART signals</a> for more details.)			
LPUART1_CTS_B / LPUART4_SIN	Clear to send	UART2_CTS_B GPIO1_22 UART4_SIN FTM4_CH5	I
LPUART1_RTS_B / LPUART4_SOUT	Request to send	UART2_RTS_B GPIO1_20 UART4_SOUT FTM4_CH3	O
LPUART1_SIN	Receive data	UART2_SIN GPIO1_18 FTM4_CH1	I
LPUART1_SOUT	Transmit data	UART2_SOUT GPIO1_16 FTM4_CH0	IO
LPUART2_CTS_B / LPUART5_SIN	Clear to send	SDHC_DAT2 GPIO2_07 FTM4_EXTCLK	I
LPUART2_RTS_B / LPUART5_SOUT	Request to send	SDHC_DAT1 GPIO2_06 FTM4_FAULT	O
LPUART2_SIN	Receive data	UART1_CTS_B GPIO1_21 UART3_SIN FTM4_CH4	I
LPUART2_SOUT	Transmit data	UART1_RTS_B GPIO1_19 UART3_SOUT FTM4_CH2	IO

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
LPUART3_CTS_B / LPUART6_SIN	Clear to send	SDHC_CLK GPIO2_09 FTM4_QD_PHB	I
LPUART3_RTS_B / LPUART6_SOUT	Request to send	SDHC_DAT3 GPIO2_08 FTM4_QD_PHA	O
LPUART3_SIN	Receive data	SDHC_DAT0 GPIO2_05 FTM4_CH7	I
LPUART3_SOUT	Transmit data	SDHC_CMD GPIO2_04 FTM4_CH6	IO
LPUART4_SIN / LPUART1_CTS_B	Receive data	UART2_CTS_B GPIO1_22 UART4_SIN FTM4_CH5	I
LPUART4_SOUT / LPUART1_RTS_B	Transmit data	UART2_RTS_B GPIO1_20 UART4_SOUT FTM4_CH3	IO
LPUART5_SIN / LPUART2_CTS_B	Receive data	SDHC_DAT2 GPIO2_07 FTM4_EXTCLK	I
LPUART5_SOUT / LPUART2_RTS_B	Transmit data	SDHC_DAT1 GPIO2_06 FTM4_FAULT	IO
LPUART6_SIN / LPUART3_CTS_B	Receive data	SDHC_CLK GPIO2_09 FTM4_QD_PHB	I
LPUART6_SOUT / LPUART3_RTS_B	Transmit data	SDHC_DAT3 GPIO2_08 FTM4_QD_PHA	IO
<b>QUICC Engine</b> (See QUICC Engine Block Reference Manual with Protocol Interworking for more details.)			
CLK10 / QE_SI1_STROBE1	QE clock	IIC2_SDA GPIO4_03 SDHC_WP FTM3_QD_PHB BRGO3	I
CLK11	QE clock	IIC3_SCL GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0	I
CLK12_CLK8	QE clock	IIC3_SDA GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		FTM8_CH1	
CLK9 / QE_SI1_STROBE0	QE clock	IIC2_SCL GPIO4_02 SDHC_CD_B FTM3_QD_PHA BRGO2	I
QE_SI1_STROBE0 / CLK9	SI strobe	IIC2_SCL GPIO4_02 SDHC_CD_B FTM3_QD_PHA BRGO2	O
QE_SI1_STROBE1 / CLK10	SI strobe	IIC2_SDA GPIO4_03 SDHC_WP FTM3_QD_PHB BRGO3	O
UC1_CDB_RXER	Receive error	IIC4_SCL GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ FTM3_FAULT	I
UC1_CTSB_RXDV	Receive data	IRQ05 GPIO1_25 FTM3_CH1 TDMA_RSYNC	I
UC1_RTSB_TXEN	Transmit enable	IRQ07 GPIO1_27 FTM3_CH3 TDMA_TSYNC	O
UC1_RXD7	Receive data	IRQ04 GPIO1_24 FTM3_CH0 TDMA_RXD TDMA_TXD	I
UC1_TXD7	Transmit data	IRQ06 GPIO1_26 FTM3_CH2 TDMA_RXD_EXC TDMA_TXD	O
UC3_CDB_RXER	Receive error	IIC4_SDA GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK	I
UC3_CTSB_RXDV	Receive data	IRQ09 GPIO1_29 FTM3_CH5 TDMB_RSYNC	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
UC3_RTSB_TXEN	Transmit enable	<b>IRQ03</b> GPIO1_23 FTM3_CH7 TDMB_TSYNC	O
UC3_RXD7	Receive data	<b>IRQ08</b> GPIO1_28 FTM3_CH4 TDMB_RXD TDMB_TXD	I
UC3_TXD7	Transmit data	<b>IRQ10</b> GPIO1_30 FTM3_CH6 TDMB_RXD_EXC TDMB_TXD	O
<b>Baud rate generator</b>			
BRGO1	Baud Rate Generator 1	<b>IIC3_SDA</b> GPIO4_11 EVT6_B USB2_PWRFAULT FTM8_CH1 CLK12_CLK8	O
BRGO2	Baud Rate Generator 2	<b>IIC2_SCL</b> GPIO4_02 SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0	O
BRGO3	Baud Rate Generator 3	<b>IIC2_SDA</b> GPIO4_03 SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1	O
BRGO4	Baud Rate Generator 4	<b>IIC3_SCL</b> GPIO4_10 EVT5_B USB2_DRVVBUS FTM8_CH0 CLK11	O
<b>Time Division Multiplexing</b> (See QUICC Engine Block Reference Manual with Protocol Interworking for more details.)			
TDMA_RQ	RQ	<b>IIC4_SCL</b> GPIO4_12 EVT7_B USB3_DRVVBUS FTM3_FAULT UC1_CDB_RXER	O
TDMA_RSYNC	RSYNC	<b>IRQ05</b> GPIO1_25 FTM3_CH1	I

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		UC1_CTSB_RXDV	
TDMA_RXD / TDMA_TXD	RXD	<b>IRQ04</b> GPIO1_24 FTM3_CH0 UC1_RXD7	I
TDMA_RXD_EXC / TDMA_TXD	Recieve Data	<b>IRQ06</b> GPIO1_26 FTM3_CH2 UC1_RXD7	I
TDMA_TSYNC	TSYNC	<b>IRQ07</b> GPIO1_27 FTM3_CH3 UC1_RTSB_TXEN	I
TDMA_TXD / TDMA_RXD	Transmit Data	<b>IRQ04</b> GPIO1_24 FTM3_CH0 UC1_RXD7	O
TDMA_TXD / TDMA_RXD_EXC	Transmit Data	<b>IRQ06</b> GPIO1_26 FTM3_CH2 UC1_RXD7	O
TDMB_RQ	RQ	<b>IIC4_SDA</b> GPIO4_13 EVT8_B USB3_PWRFAULT FTM3_EXTCLK UC3_CDB_RXER	O
TDMB_RSYNC	RSYNC	<b>IRQ09</b> GPIO1_29 FTM3_CH5 UC3_CTSB_RXDV	I
TDMB_RXD / TDMB_TXD	RXD	<b>IRQ08</b> GPIO1_28 FTM3_CH4 UC3_RXD7	I
TDMB_RXD_EXC / TDMB_TXD	Recieve Data	<b>IRQ10</b> GPIO1_30 FTM3_CH6 UC3_RXD7	I
TDMB_TSYNC	TSYNC	<b>IRQ03</b> GPIO1_23 FTM3_CH7 UC3_RTSB_TXEN	I
TDMB_TXD / TDMB_RXD	Transmit Data	<b>IRQ08</b> GPIO1_28 FTM3_CH4 UC3_RXD7	O
TDMB_TXD / TDMB_RXD_EXC	Transmit Data	<b>IRQ10</b> GPIO1_30 FTM3_CH6	O

*Table continues on the next page...*

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		UC3_TXD7	
<b>TSEC_1588</b> (See DPAA reference manual for more details.)			
TSEC_1588_ALARM_O_UT1	Alarm Out	<b>EC2_TXD2</b> GPIO3_16 FTM2_CH7	O
TSEC_1588_ALARM_O_UT2	Alarm Out	<b>EC2_TXD3</b> GPIO3_15 FTM2_CH5	O
TSEC_1588_CLK_IN	Clock In	<b>EC2_RX_CLK</b> GPIO3_26 FTM2_QD_PHA	I
TSEC_1588_CLK_OUT	Clock Out	<b>EC2_TXD1</b> GPIO3_17 FTM2_CH3	O
TSEC_1588_PULSE_OU_T1	Pulse Out	<b>EC2_RXD1</b> GPIO3_24 FTM2_CH1	O
TSEC_1588_PULSE_OU_T2	Pulse Out	<b>EC2_TXD0</b> GPIO3_18 FTM2_CH2	O
TSEC_1588_TRIG_IN1	Trigger In	<b>EC2_RX_DV</b> GPIO3_27 FTM2_QD_PHB	I
TSEC_1588_TRIG_IN2	Trigger In	<b>EC2_RXD0</b> GPIO3_25 FTM2_CH0	I
<b>QSPI</b> (See <a href="#">LS1043A QuadSPI signals</a> for more details.)			
QSPI_A_CS0	Chip Select	<b>IFC_A16</b>	O
QSPI_A_CS1	Chip Select	<b>IFC_A17</b>	O
QSPI_A_DATA0	Data	<b>IFC_A22</b> IFC_WP1_B	IO
QSPI_A_DATA1	Data	<b>IFC_A23</b> IFC_WP2_B	IO
QSPI_A_DATA2	Data	<b>IFC_A24</b> IFC_WP3_B	IO
QSPI_A_DATA3	Data	<b>IFC_A25</b> GPIO2_25 FTM5_CH0 IFC_CS4_B IFC_RB2_B	IO
QSPI_A_SCK	QSPI_A Clock	<b>IFC_A18</b>	O
QSPI_B_CS0	Chip Select	<b>IFC_A19</b>	O
QSPI_B_CS1	Chip Select	<b>IFC_A20</b>	O
QSPI_B_DATA0	Data	<b>IFC_PAR0</b> GPIO2_13 FTM6_CH0	IO

Table continues on the next page...

**Table 3-1. LS1043 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
QSPI_B_DATA1	Data	IFC_PAR1 GPIO2_14 FTM6_CH1	IO
QSPI_B_DATA2	Data	IFC_PERR_B GPIO2_15 FTM6_EXTCLK	IO
QSPI_B_DATA3	Data	IFC_CS3_B GPIO2_12 FTM7_EXTCLK	IO
QSPI_B_DATA3	Data	IFC_CS3_B GPIO2_12 FTM7_EXTCLK	IO
QSPI_B_SCK	QSPI_B Clock	IFC_A21 cfg_dram_type	O

### 3.3 Configuration signals sampled at reset

The signals that serve alternate functions as configuration input signals during system reset are summarized in this table.

Note that throughout this document, the reset configuration signals are described as being sampled at the negation of PORESET\_B. However, there is a setup and hold time for these signals relative to the rising edge of PORESET\_B, as described in the chip's data sheet document.

The reset configuration signals are multiplexed with other functional signals. The values on these signals during reset are interpreted to be logic one or zero, regardless of whether the functional signal name is defined as active-low. The reset configuration signals have internal pull-up resistors so that if the signals are not driven, the default value is high (a one), as shown in the table. Some signals must be driven high or low during the reset period. For details about all the signals that require external pull-up resistors, see the data sheet document.

**Table 3-2. Reset configuration signals**

Reset configuration name	Functional interface	Functional Signal Name	Default
cfg_rcw_src[0:7]	IFC	IFC_AD[8:15]	1111 1111
cfg_rcw_src[8]	IFC	IFC_CLE	1 <sup>1</sup>
cfg_ifc_te	IFC	IFC_TE	1
cfg_dram_type	IFC	IFC_A[21]	1

*Table continues on the next page...*

**Table 3-2. Reset configuration signals (continued)**

Reset configuration name	Functional interface	Functional Signal Name	Default
cfg_gpininput[0:7]	IFC	IFC_AD[0:7]	1111 1111
cfg_eng_use0	IFC	IFC_WE0_B	1
cfg_eng_use1	IFC	IFC_OE_B	1
cfg_eng_use2	IFC	IFC_WP0_B	1

1. cfg\_rcw\_src[0:8]=1\_1111\_1111 is not a valid setting; They must be set to one of the valid options defined in [Reset configuration word \(RCW\) source](#)

## 3.4 Signal multiplexing details

Due to the extensive functionality present on the chip and the limited number of external signals available, several functional blocks share signal resources through multiplexing. These signals are designated in the Alternate function(s) column of [Table 3-1](#). In this case when there is alternate functionality between multiple functional blocks, the signal's function is determined at the chip level (rather than at the block level) typically by a reset configuration word (RCW) option. For example, the signal SPI\_CS\_B[0:3]/SDHC\_DAT[4:7]/GPIO2[0:3] can be utilized by either the SPI block, the eSDHC block, or by GPIO2. Because these signals alternatively service three different functional blocks, their function is determined at the chip level by RCW[SPI\_BASE]. The following sections describe external signal functional selection using the RCW.

### 3.4.1 UART, GPIO, FTM, and LPUART signal multiplexing

The functionality of these signals is determined by the RCW[UART\_BASE] and RCW[UART\_EXT]).

**Table 3-3. UART signal configuration**

Signal name	Signal function	RCW[UART_BASE]	RCW[UART_EXT]
UART1_SOUT	GPIO1[15]	0h	0xx
	UART1_SOUT	3h, 4h, 5h, 6h, 7h	
UART1_SIN	GPIO1[17]	0h	0xx
	UART1_SIN	3h, 4h, 5h, 6h, 7h	
UART1_RTS_B	GPIO1[19]	0h, 3h, 5h	000
	UART1_RTS_B	4h, 6h	
	UART3_SOUT	7h	
	LPUART2_SOUT	Don't care	001/ 010
	FTM4_CH2	Don't care	011

Table continues on the next page...

**Table 3-3. UART signal configuration (continued)**

Signal name	Signal function	RCW[UART_BASE]	RCW[UART_EXT]
UART1_CTS_B	GPIO1[21]	0h, 3h, 5h	000
	UART1_CTS_B	4h, 6h	
	UART3_SIN	7h	
	LPUART2_SIN	Don't care	001/010
	FTM4_CH4	Don't care	011
UART2_SOUT	GPIO1[16]	0h, 3h, 4h	000
	UART2_SOUT	6h, 5h, 7h	
	LPUART1_SOUT	Don't care	001/010
	FTM4_CH0	Don't care	011
UART2_SIN	GPIO1[18]	0h, 3h, 4h	000
	UART2_SIN	6h, 5h, 7h	
	LPUART1_SIN	Don't care	001/010
	FTM4_CH1	Don't care	011
UART2_RTS_B	GPIO1[20]	0h, 3h, 4h, 5h	000
	UART2_RTS_B	6h	
	UART4_SOUT	7h	
	LPUART1_RTS_B	Don't care	001
	LPUART4_SOUT		010
	FTM4_CH3		011
UART2_CTS_B	GPIO1[22]	0h, 3h, 4h, 5h	000
	UART2_CTS_B	6h	000
	UART4_SIN	7h	000
	LPUART1_CTS_B	Don't care	001
	LPUART4_SIN		010
	FTM4_CH5		011

### 3.4.2 ASLEEP and GPIO1 signal multiplexing

The power management signal, ASLEEP, is shared with GPO1[13]. The functionality of this signal is determined by the ASLEEP field in the reset configuration word (RCW[ASLEEP]).

**Table 3-4. ASLEEP signal configuration**

Signal name	Signal function	RCW[ASLEEP]
ASLEEP	ASLEEP	0
	GPO1[13]	1

### 3.4.3 RTC and GPIO1 signal multiplexing

The clocking signal, RTC, is shared with GPIO1[14]. The functionality of this signal is determined by the RTC field in the reset configuration word (RCW[RTC]).

**Table 3-5. RTC signal configuration**

Signal name	Signal function	RCW[RTC]
RTC	RTC	0
	GPIO1[14]	1

### 3.4.4 eSDHC, GPIO2, and GPIO4 signal multiplexing

The eSDHC interface shares signals with GPIO2 and GPIO4. The functionality of these signals are determined by the SDHC\_BASE and SDHC fields in the reset configuration word (RCW[SDHC\_BASE] and RCW[SDHC]).

**Table 3-6. eSDHC BASE signal configuration**

Signal name	Signal function	RCW[SDHC_BASE]	RCW[SDHC_EXT]
SDHC_CMD	SDHC_CMD	0	000
	GPIO2[4]	1	
	LPUART3_S OUT	Don't care	001/010
	FTM4_CH6		011
SDHC_DAT[0]	SDHC_DAT[0]	0	000
	GPIO2[5]	1	
	LPUART3_SI N	Don't care	001/010
	FTM4_CH7		011
SDHC_DAT[1]	SDHC_DAT[1]	0	000
	GPIO2[6]	1	
	LPUART2_R TS_B	Don't care	001
	LPUART5_S OUT		010
	FTM4_FAULT		011
SDHC_DAT[2]	SDHC_DAT[2]	0	000

*Table continues on the next page...*

**Table 3-6. eSDHC BASE signal configuration  
(continued)**

Signal name	Signal function	RCW[SDHC_BASE]	RCW[SDHC_EXT]
	GPIO2[7]	1	
	LPUART2_C_TS_B	Don't care	001
	LPUART5_SI_N		010
	FTM4_EXTC_LK		011
SDHC_DAT[3]	SDHC_DAT[3]	0	000
	GPIO2[8]	1	
	LPUART3_R_TS_B	Don't care	001
	LPUART6_S_OUT		010
	FTM4_QD_PHA		011
SDHC_CLK	SDHC_CLK	0	000
	GPIO2[9]	1	
	LPUART3_C_TS_B	Don't care	001
	LPUART6_SI_N		010
	FTM4_QD_PHB		011

### 3.4.5 External IRQ, QE, and GPIO1 signal multiplexing

External IRQ share signals with GPIO1. The functionality of these signals is determined by the IRQ\_BASE and IRQ\_EXT fields in the reset configuration word (RCW[IRQ\_BASE] and RCW[IRQ\_EXT]).

**Table 3-7. IRQn signal configuration**

Signal name	Signal function	RCW[IRQ_BASE]	RCW[IRQ_EXT]
IRQ[3]	IRQ[3]	0	000
	GPIO1[23]	1	
	TDMB_TSYNC	Don't care	001
	UC3_RTSB_TXEN		010
	FTM3_CH7		011

*Table continues on the next page...*

**Table 3-7. IRQn signal configuration (continued)**

Signal name	Signal function	RCW[IRQ_BASE]	RCW[IRQ_EXT]
IRQ[4]	IRQ[4]	0	000
	GPIO1[24]	1	
	TDMA_TXD/TDMA_RXD	Don't care	001
	UC1_RXD[7]		010
	FTM3_CH0		011
IRQ[5]	IRQ[5]	0	000
	GPIO1[25]	1	
	TDMA_RSYNC	Don't care	001
	UC1_CTSB_RXDV		010
	FTM3_CH1		011
IRQ[6]	IRQ[6]	0	000
	GPIO1[26]	1	
	TDMA_TXD/TDMA_RXD_EXC	Don't care	001
	UC1_TXD[7]		010
	FTM3_CH2		011
IRQ[7]	IRQ[7]	0	000
	GPIO1[27]	1	
	TDMA_TSYNC	Don't care	001
	UC1_RTSB_TXEN		010
	FTM3_CH3		011
IRQ[8]	IRQ[8]	0	000
	GPIO1[28]	1	
	TDMB_TXD/TDMD_RXD	Don't care	001
	UC3_RXD[7]		010
	FTM3_CH4		011
IRQ[9]	IRQ[9]	0	000
	GPIO1[29]	1	
	TDMD_RSYNC	Don't care	001
	UC3_CTSB_RXDV		010
	FTM3_CH5		011
IRQ[10]	IRQ[10]	0	000
	GPIO1[30]	1	
	TDMD_TXD/TDMD_RXD_EXC	Don't care	001
	UC3_TXD[7]		010
	FTM3_CH6		011
IRQ[11]	IRQ[11]	0	000/011
	GPIO1[31]	1	

### 3.4.6 SPI, eSDHC, USB and GPIO2 signal multiplexing

The SPI share signals with the eSDHC and GPIO2. The functionality of these signals is determined by the SPI\_BASE and SPI\_EXT fields in the reset configuration word (RCW[SPI\_BASE] and RCW[SPI\_EXT]).

**Table 3-8. SPI Signal configuration**

Signal name	Signal function	RCW[SPI_BASE]	RCW[SPI_EXT]
SPI_CS_B[0]	SPI_CS_B[0]	00	000/010
	SDHC_DAT[4]	01	
	GPIO2[0]	10	
	Reserved	11	
	SDHC_VS	Don't care	001
SPI_CS_B[1]	SPI_CS_B[1]	00	000/001/010
	SDHC_DAT[5]	01	
	GPIO2[1]	10	
	SDHC_CMD_DIR	11	
SPI_CS_B[2]	SPI_CS_B[2]	00	000/001/010
	SDHC_DAT[6]	01	
	GPIO2[2]	10	
	SDHC_DAT0_DIR	11	
SPI_CS_B[3]	SPI_CS_B[3]	00	000/001/010
	SDHC_DAT[7]	01	
	GPIO2[3]	10	
	SDHC_DAT123_DIR	11	
SPI_MOSI	SPI_MOSI	00	000/010
	SDHC_CLK_SYNC_OUT	Don't care	001
SPI_MISO	SPI_MISO	00	000/010
	SDHC_CLK_SYNC_IN	Don't care	001
SPI_CLK	SPI_CLK	00	000
	Reserved	Don't care	001
	Reserved		010

#### NOTE

- 8 bit MMC DDR is not supported.
- SPI\_CLK is available only when RCW[SPI\_EXT]=000 and RCW[SPI\_BASE]=00. Some SPI signals are also available when RCW[SPI\_EXT]=010, however SPI\_CLK must be generated by master.

### 3.4.7 IFC, QuadSPI, FTM and GPIO2 signal multiplexing

The integrated Flash controller shares signals with QuadSPI, FTM and GPIO2. The functionality of these signals is determined by the IFC\_GRP\_[n]\_BASE fields and IFC\_GRP\_[m]\_EXT in the reset configuration word as described in the following table.

**Table 3-9. IFC signal configuration**

Signal name	Signal function	IFC_A_22_24	IFC_GRP_[n]_BASE			IFC_GRP_[m]_EXT			
			E1	D	A	F	E1	D	A
IFC_A[16]	IFC_A[16]	-	-	-	-	000			
	QSPI_A_C_S0	-	-	-	-	001			
IFC_A[17]	IFC_A[17]	-	-	-	-	000			
	QSPI_A_C_S1	-	-	-	-	001			
IFC_A[18]	IFC_A[18]	-	-	-	-	000			
	QSPI_A_S_CK	-	-	-	-	001			-
IFC_A[19]	IFC_A[19]	-	-	-	-	000			
	QSPI_B_C_S0	-	-	-	-	001			
IFC_A[20]	IFC_A[20]	-	-	-	-	000			
	QSPI_B_C_S1	-	-	-	-	001			
IFC_A[21]	IFC_A[21]	-	-	-	-	000			
	QSPI_B_S_CK	-	-	-	-	001			
IFC_A[22]	IFC_A[22]	0		-	-	000			
	IFC_WP_B[1]			-	-	000			
	QSPI_A_D_ATA[0]	Don't care		-	-	001			
IFC_A[23]	IFC_A[23]	0		-	-	000			
	IFC_WP_B[2]			-	-	000			
	QSPI_A_D_ATA[1]	Don't care		-	-	001			
IFC_A[24]	IFC_A[24]	0		-	-	000			
	IFC_WP_B[3]			-	-	000			
	QSPI_A_D_ATA[2]	Don't care		-	-	001			
IFC_A[25]	IFC_A[25]	-	-	-	00	-	-	-	000
	GPIO2[25]	-	-	-	01	-	-	-	000

Table continues on the next page...

**Table 3-9. IFC signal configuration (continued)**

	IFC_RB_B[2]	-	-	-	10	-	-	-	000
	Reserved	-	-	-	11	-	-	-	000
	QSPI_A_D ATA[3]	-	-	-	Don't care	-	-	-	001
	FTM5_CH0	-	-	-	Don't care	-	-	-	010
	IFC_CS_B[4]	-	-	-	Don't care	-	-	-	100
IFC_A[26]	IFC_A[26]	-	-	-	00	-	-	-	000
	GPIO2[26]	-	-	-	01	-	-	-	000
	IFC_RB_B[3]	-	-	-	10	-	-	-	000
	Reserved	-	-	-	11	-	-	-	000
	QSPI_A_D QS	-	-	-	Don't care	-	-	-	001
	FTM5_CH1	-	-	-	Don't care	-	-	-	010
	IFC_CS_B[5]	-	-	-	Don't care	-	-	-	100
IFC_A[27]	IFC_A[27]	-	-	-	00	-	-	-	000
	GPIO2[27]	-	-	-	01	-	-	-	000
	QSPI_B_D QS	-	-	-	-	-	-	-	001
	FTM5_EXT CLK	-	-	-	-	-	-	-	010
	IFC_CS_B[6]	-	-	-	-	-	-	-	100
IFC_PAR[0]	IFC_PAR[0]	-	-	0	-	-	-	000	-
	GPIO2[13]	-	-	1	-	-	-	-	-
	QSPI_B_D ATA[0]	-	-	Don't care	-	-	-	001	-
	FTM6_CH0	-	-	Don't care	-	-	-	010	-
IFC_PAR[1]	IFC_PAR[1]	-	-	0	-	-	-	000	-
	GPIO2[14]	-	-	1	-	-	-	000	-
	QSPI_B_D ATA[1]	-	-	Don't care	-	-	-	001	-
	FTM6_CH1	-	-	Don't care	-	-	-	010	-
IFC_CS_B[1]	IFC_CS_B[1]	-	0	-	-	-	000/001	-	-
	GPIO2[10]	-	1	-	-	-	-	-	-
	FTM7_CH0	-	Don't care	-	-	-	010	-	-
IFC_CS_B[2]	IFC_CS_B[2]	-	0	-	-	-	000/001	-	-
	GPIO2[11]	-	1	-	-	-	-	-	-
	FTM7_CH1	-	Don't care	-	-	-	010	-	-

Table continues on the next page...

**Table 3-9. IFC signal configuration (continued)**

IFC_CS_B[3]	IFC_CS_B[3]	-	0	-	-	-	000	-	-
	GPIO2[12]	-	1	-	-	-	000	-	-
	QSPI_B_D ATA[3]	-	Don't care	-	-	-	001	-	-
	FTM7_EXT CLK	-		-	-	-	010	-	-
IFC_PERR_B	IFC_PERR_B	-	-	0	-	-	-	000	-
	GPIO2[15]	-	-	1	-	-			-
	QSPI_B_D ATA[2]	-	-	Don't care	-	-	-	001	-
	FTM6_EXT CLK	-	-		-	-	-	010	-

### 3.4.8 Ethernet controller 1, FTM1, and GPIO3 signal multiplexing

The Ethernet controller 1 (EC1) RGMII interface shares signals with FTM and GPIO3. The functionality of these signals is determined by the EC1 field in the reset configuration word (RCW[EC1]).

**Table 3-10. EC1 RGMII signal configuration**

Signal name	Signal function	RCW[EC1]
EC1_TXD[3:0]	EC1_TXD[3:0]	000
	GPIO3[2:5]	001
	FTM1_CH5, FTM1_CH7, FTM1_CH3, FTM1_CH2	101
EC1_TX_EN	EC1_TX_EN	000
	GPIO3[6]	001
	FTM1_FAULT	101
EC1_GTX_CLK	EC1_GTX_CLK	000
	GPIO3[7]	001
	FTM1_EXT_CLK	101
EC1_GTX_CLK125	EC1_GTX_CLK125	000
	GPIO3[8]	001
EC1_RXD[3:0]	EC1_RXD[3:0]	000
	GPIO3[9:12]	001
	FTM1_CH4, FTM1_CH6, FTM1_CH1, FTM1_CH0	101
EC1_RX_CLK	EC1_RX_CLK	000
	GPIO3[13]	001

Table continues on the next page...

**Table 3-10. EC1 RGMII signal configuration (continued)**

Signal name	Signal function	RCW[EC1]
	FTM1_QD_PHA	101
EC1_RX_DV	EC1_RX_DV	000
	GPIO3[14]	001
	FTM1_QD_PHB	101

### 3.4.9 Ethernet controller 2, GPIO3, FTM2, and IEEE1588 signal multiplexing

The Ethernet controller 2 (EC2) RGMII interface shares signals with GPIO3 and GPIO4. The functionality of these signals is determined by the EC2 field in the reset configuration word (RCW[EC2]).

**Table 3-11. EC2 RGMII signal configuration**

Signal name	Signal function	RCW[EC2]
EC2_TXD[3:0]	EC2_TXD[3:0]	000
	GPIO3[15:18]	001
	TSEC_1588_ALARM_OUT[2:1], TSEC_1588_CLK_OUT, TSEC_1588_PULSE_OUT2	010
	FTM2_CH5, FTM2_CH7, FTM2_CH3, FTM2_CH2	101
EC2_TX_EN	EC2_TX_EN	000
	GPIO3[19]	001, 010
	FTM2_FAULT	101
EC2_GTX_CLK	EC2_GTX_CLK	000
	GPIO3[20]	001, 010
	FTM2_EXTCLK	101
EC2_GTX_CLK125	EC2_GTX_CLK125	000
	GPIO3[21]	001, 010
EC2_RXD[3:2]	EC2_RXD[3:2]	000
	GPIO3[22:23]	001, 010
	FTM2_CH4, FTM2_CH6	101
EC2_RXD[1]	EC2_RXD[1]	000
	GPIO3[24]	001
	TSEC_1588_PULSE_OUT1	010
	FTM2_CH1	101
EC2_RXD[0]	EC2_RXD[0]	000
	GPIO3[25]	001
	FTM2_CH0	101

Table continues on the next page...

**Table 3-11. EC2 RGMII signal configuration (continued)**

Signal name	Signal function	RCW[EC2]
	TSEC1588_TRIG_IN2	010
EC2_RX_DV	EC2_RX_DV	000
	GPIO3[27]	001
	TSEC1588_TRIG_IN1	010
	FTM2_QD_PHB	101
EC2_RX_CLK	EC2_RX_CLK	000
	GPIO3[26]	001
	TSEC_1588_CLK_IN	010
	FTM2_QD_PHA	101

### 3.4.10 Ethernet management interface1 and GPIO3 signal multiplexing

The Ethernet management interface1 (EMI1) shares signals with GPIO3. The functionality of these signals is determined by the EM1 field in the reset configuration word (RCW[EM1]).

**Table 3-12. EC1 RGMII signal configuration**

Signal name	Signal function	RCW[EM1]
EMI1_MDC	EMI1_MDC	0
	GPIO_3[0]	1
EMI1_MDIO	EMI1_MDIO	0
	GPIO_3[1]	1

### 3.4.11 Ethernet management interface2 and GPIO4 signal multiplexing

The Ethernet management interface 2 (EMI2) shares signals with GPIO4. The functionality of these signals is determined by the EM2 field in the reset configuration word (RCW[EM2]).

**Table 3-13. EC2 RGMII signal configuration**

Signal name	Signal function	RCW[EM2]
EMI2_MDC	EMI1_MDC	0
	GPIO_4[0]	1

*Table continues on the next page...*

**Table 3-13. EC2 RGMII signal configuration (continued)**

Signal name	Signal function	RCW[EM2]
EMI2_MDIO	EMI1_MDIO	0
	GPIO_4[1]	1

### 3.4.12 I<sup>2</sup>C2, GPIO4, FTM, QE and eSDHC signal multiplexing

The I<sup>2</sup>C2 interface shares signals with GPIO4 and esDHC. The functionality of these signals is determined by the RCW[I<sup>2</sup>C2\_EXT].

**Table 3-14. I<sup>2</sup>C2 signal configuration**

Signal name	Signal function	RCW[I <sup>2</sup> C2_EXT]
IIC2_SCL	IIC2_SCL	000
	SDHC_CD_B	001
	GPIO_4[2]	010
	FTM3_QD_PHA	011
	QE_SI1_STROBE[0]	100
	CLK9	101
	BRGO2	110
IIC2_SDA	IIC2_SDA	000
	SDHC_WP	001
	GPIO_4[3]	010
	FTM3_QD_PHB	011
	QE_SI1_STROBE[1]	100
	CLK10	101
	BRGO3	110

### 3.4.13 USB and GPIO4 signal multiplexing

The USB (USB\_DRVVBUS/USB\_PWRFAULT) shares signals with GPIO4. The functionality of these signals is determined by the reset configuration word (RCW).

**Table 3-15. USB signal configuration**

Signal name	Signal function	RCW[USB_DRVVBUS]/ RCW[USB_PWRFAULT]
USB_DRVVBUS	USB_DRVVBUS	0
	GPIO4[29]	1

*Table continues on the next page...*

**Table 3-15. USB signal configuration  
(continued)**

Signal name	Signal function	RCW[USB_DRVVBUS]/ RCW[USB_PWRFAULT]
USB_PWRFAULT	USB_PWRFAULT	0
	GPIO4[30]	1

## 3.5 Output Signal States During Reset

When a system reset is initiated (HRESET\_B or PORESET\_B sampled asserted by the chip), the chip aborts all current internal and external transactions and releases the bidirectional I/O signals to a high-impedance state. However, some signals get stable values at power-on reset.

While the the chip is in reset, it drives HRESET\_B asserted and ignores most input signals (except for the reset configuration signals) and drives output-only signal MODT[0:1] to an inactive state. Signal MCKE[0:1] is driven to an inactive state after PORESET\_B is de-asserted.

Note that signals associated with all potential RCW source interfaces are enabled and active while the chip is in reset (that is, while HRESET\_B is being driven asserted by the chip, but after PORESET\_B is deasserted). This is necessary to allow the interfaces to be used for fetching configuration information from non-volatile memory devices.



# Chapter 4

## Reset, Clocking, and Initialization

### 4.1 Reset, clocking, and initialization overview

This content describes the reset, clocking, and initialization, including a definition of the reset configuration signals and the options they select. Note that other chapters in this book may describe specific aspects of initialization for individual blocks.

The reset, clocking, and control signals provide many options for operation. Additionally, many modes are selected with reset configuration signals during a power on reset (assertion of PORESET\_B) and by using the reset configuration word (RCW) functionality.

### 4.2 External signal descriptions

The table below summarizes the external signals described in this chapter.

[Table 4-3](#) and [Table 4-4](#) have detailed signal descriptions, but this table contains references to additional sections that contain more information.

**Table 4-1. Reset and Control Signals Summary**

Signal	I/O	Description	References
PORESET_B	I	Power on reset input.	<a href="#">Table 4-3</a>
HRESET_B	I/O	Hard reset input. Functions as an output during initial steps in the power on reset sequence. See <a href="#">Power-on reset sequence</a> for more information.	<a href="#">Table 4-3</a>
RESET_REQ_B	O	Reset request output. An internal block requests that either HRESET_B or PORESET_B be asserted.	<a href="#">Table 4-3</a>
CKSTP_OUT_B	O	Checkstop out.	<a href="#">Table 4-3</a>

*Table continues on the next page...*

**Table 4-1. Reset and Control Signals Summary (continued)**

Signal	I/O	Description	References
		This signal is reserved for internal use. Refer the chip specific design checklist for more information.	
ASLEEP	O	Asleep	<a href="#">Table 4-3</a>
TA_TMP_DETECT_B	I	Tamper Detect.	<a href="#">Table 4-3</a>
TA_BB_TMP_DETECT_B	I	Low Power Tamper Detect.	<a href="#">Table 4-3</a>

**Table 4-2. Clock Signals Summary**

Signal	I/O	Description	References
SYSCLK	I	Primary clock input.	<a href="#">Table 4-4</a>
DIFF_SYSCLK	I	Single Oscillator Source Clock Differential (positive)	<a href="#">Table 4-4</a>
DIFF_SYSCLK_B	I	Single Oscillator Source Clock Differential (positive)	<a href="#">Table 4-4</a>
RTC	I	Real time clock input.	<a href="#">Table 4-4</a>
TA_BB_RTC	I	Low-power real time clock. This signal is reserved for internal use. Refer the chip specific design checklist for more information.	<a href="#">Table 4-4</a>
SD_REF_CLKn_P/ SD_REF_CLKn_N	I	SerDes high-speed interface reference clock $n$ .	<a href="#">Table 4-4</a>
CLK_OUT	O	Diagnostic clock output.	<a href="#">Table 4-4</a>
DDRCLK	I	Reference clock for DDR controller.	<a href="#">Table 4-4</a>

The following sections describe the reset and clock signals in detail.

### 4.2.1 System control signals

The table below describes some of the system control signals. [Power-on reset configuration](#) describes the signals that also function as reset configuration signals.

**Table 4-3. System control signals: Detailed signal descriptions**

Signal	I/O	Description
PORESET_B	I	Power on reset. Causes the chip to abort all current internal and external transactions and set all registers to their default values. PORESET_B may be asserted completely asynchronously with respect to all other signals.

*Table continues on the next page...*

**Table 4-3. System control signals: Detailed signal descriptions (continued)**

Signal	I/O	Description	
		<b>State Meaning</b>	Asserted/Negated-See <a href="#">Signal Descriptions</a> and <a href="#">Power-on reset configuration</a> for more information on the interpretation of the other signals during reset.
		<b>Timing</b>	Assertion/Negation-The chip data sheet gives specific timing information for this signal and the reset configuration signals.
HRESET_B	I/O		Hard reset. Causes the chip to abort all current internal and external transactions and set all registers to their default values. HRESET_B may be asserted completely asynchronously with respect to all other signals. HRESET_B is driven as an output during the first part of the power on reset sequence, after which, it becomes an input, allowing external devices to stall/hold the reset sequence. See <a href="#">Hard reset sequence</a> for more information.
		<b>State Meaning</b>	Asserted/Negated-See and <a href="#">Power-on reset configuration</a> for more information on the interpretation of the other signals during reset.
		<b>Timing</b>	Assertion/Negation-The chip data sheet gives specific timing information for this signal.
RESET_REQ_B	O		Reset request. Indicates to the board (system in which the chip is embedded) that a condition requiring the assertion of HRESET_B or PORESET_B has been detected.
		<b>State Meaning</b>	Asserted-An event has triggered a request for either a hard reset or a power on reset. See <a href="#">Reset Request Status Register (DCFG_CCSR_RSTRQSR1)</a> Negated-Indicates no reset request.
		<b>Timing</b>	Assertion/Negation-May occur any time. Once asserted, RESET_REQ_B does not negate until either HRESET_B or PORESET_B is asserted.
CKSTP_OUT_B	O		Checkstop out. Note that in LS1043A chip this signal is reserved for internal use. For more information, refer the chip specific design checklist.
		<b>State Meaning</b>	Asserted-Indicates that the chip is in a checkstop state. Negated-Indicates normal operation. After CKSTP_OUT_B has been asserted, it is negated after the next negation (low-to-high transition) of PORESET_B.
		<b>Timing</b>	Assertion-May occur at any time; may be asserted asynchronously to the input clocks. Negation-Must remain asserted until the chip has been reset with a PORESET_B.
ASLEEP	O	Power Management Signal. See <a href="#">External Signal Description</a>	
TA_TMP_DETECT_B	I	Tamper Detect.	
TA_BB_TMP_DETECT_B	I	Low Power Tamper Detect.	

## 4.2.2 External clock signals

The table below describes some of the external clock signals of the chip. Note that some clock signals are specific to modules within the chip, and although some of their functionality is described here, they are defined in detail in their respective chapters.

**Table 4-4. Clock External Signals-Detailed Signal Descriptions**

Signal	I/O	Description	
SYSCLK	I	System clock (SYSCLK). SYSCLK is the primary clock input to the chip.	
		Timing	Assertion/Negation-See the chip data sheet for specific timing information for this signal.
DIFF_SYSCLK, DIFF_SYSCLK_B	I	Differential system clock positive terminal /Differential system clock negative terminal. These signals provide the clock to System Clock associated PLL's and if programmed to DDR PLL.	
		Timing	Assertion/Negation-See the chip data sheet for specific timing information for this signal.
RTC	I	Real time clock. The RTC timing specifications are given in the chip data sheet .	
		Timing	Assertion/Negation-See the chip data sheet for specific timing information for this signal.
TA_BB_RTC	I	This signal is reserved for internal use.	
		Timing	Assertion/Negation-Refer the chip specific design checklist for more information.
SD_REF_CLKn_P, SD_REF_CLKn_N	I	SerDes high-speed interface differential reference clocks. These differential clock inputs are used to independently clock the banks/ports of high-speed differential signal lanes available on the chip. The SerDes reference clock timing specifications are given in the chip data sheet . See the Valid reference clocks and PLL configurations for SerDes protocols in the SerDes module chapter.	
		Timing	Assertion/Negation-See the chip data sheet for specific timing information for these signals.
CLK_OUT	O	Diagnostic clock output. This output may be configured to offer one of a variety of internal system clocks to external hardware for diagnostic or debug purposes. See <a href="#">CLK_OUT configuration</a> .	
DDRCLK	I	DDR controller complex clock. DDRCLK is the reference clock to the DDR PLL.	

## 4.3 Clocking Memory Map

The following table summarizes the memory mapped registers which are used to configure clocking features.

**Clocking memory map**

Offset address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
1EE_1000	Core cluster n clock control/status register (Clocking_CLKCCSR)	32	R/W	0000_0000h	<a href="#">4.3.1/127</a>
1EE_1010	Clock generator n hardware accelerator control/status register (Clocking_CL1KCGHWACSR)	32	R/W	0000_0000h	<a href="#">4.3.2/128</a>
1EE_1030	Clock generator n hardware accelerator control/status register (Clocking_CL2KCGHWACSR)	32	R/W	0000_0000h	<a href="#">4.3.2/128</a>
1EE_1800	PLL cluster n general status register (Clocking_PLLC1GSR)	32	R	0000_0000h	<a href="#">4.3.3/130</a>
1EE_1820	PLL cluster n general status register (Clocking_PLLC2GSR)	32	R	0000_0000h	<a href="#">4.3.3/130</a>
1EE_1A00	Platform clock domain control/status register (Clocking_CLKPCSR)	32	R/W	0000_0000h	<a href="#">4.3.4/132</a>
1EE_1C00	Platform PLL general status register (Clocking_PLLPGSR)	32	R/W	0000_0000h	<a href="#">4.3.5/133</a>
1EE_1C20	DDR PLL general status register (Clocking_PLLDGSR)	32	R/W	0000_0000h	<a href="#">4.3.6/135</a>

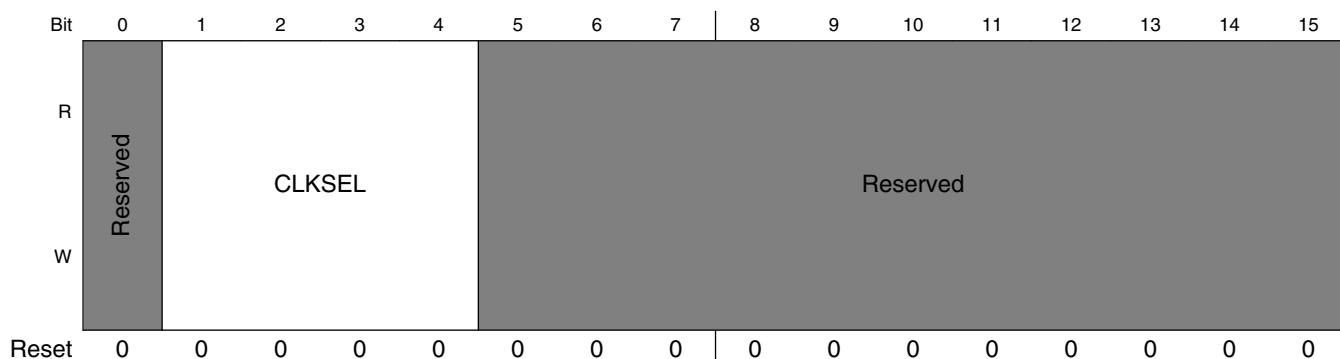
### 4.3.1 Core cluster n clock control/status register (Clocking\_CLKCCSR)

The CLKCnCSR registers control the clock frequency selection for each core cluster.

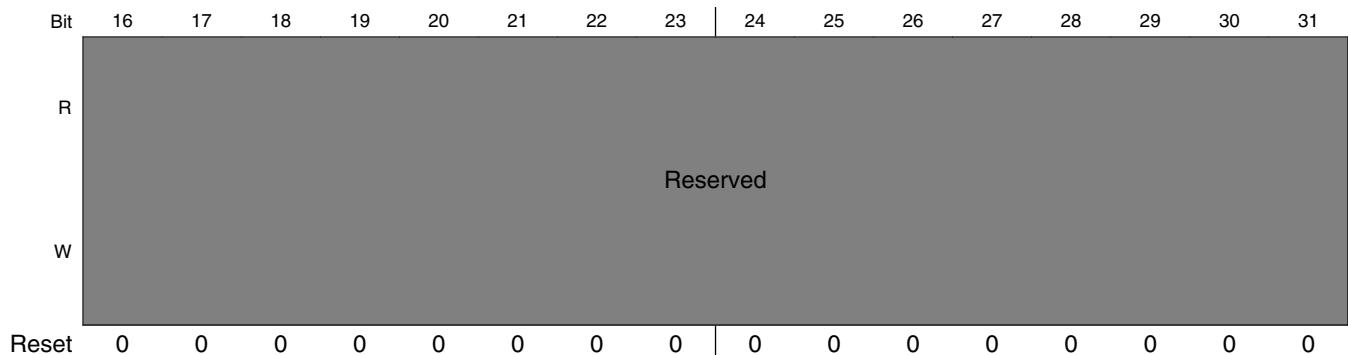
**Table 4-5. Core cluster assignments to CLKCnCSR registers**

Register	Cluster Group	Core Cluster
CLKC1CSR	Cluster Group A (CGA)	Core Cluster 1

Address: 1EE\_1000h base + 0h offset = 1EE\_1000h



## Clocking Memory Map



### Clocking\_CLKCCSR field descriptions

Field	Description
0 -	This field is reserved. Reserved
1–4 CLKSEL	Clock Select. Selects the clock source for the corresponding core cluster. See <a href="#">Table 4-5</a> above.  0000 Corresponding cluster group PLL1 output 0001 Corresponding cluster group PLL1 output divide-by-2 0010 Reserved 0100 Corresponding cluster group PLL2 output 0101 Corresponding cluster group PLL2 output divide-by-2 all others Reserved
5–31 -	This field is reserved. Reserved

### 4.3.2 Clock generator n hardware accelerator control/status register (Clocking\_CLnKCGHWACSR)

The CLKCGnHWACSR registers control the clock frequency selection for each hardware accelerator.

**Table 4-6. Hardware accelerator assignments to CLKCGnHWACSR registers**

Register	Cluster Group	Mux	Hardware Accelerator
CLKCG1HWACSR	CGA	M1	FMan  Refer RCW[HWA_CGA_M1_CLK_S EL] for more information.
CLKCG2HWACSR	CGA	M2	eSDHC  QuadSPI  Refer RCW[HWA_CGA_M2_CLK_S EL] for more information.

Address: 1EE\_1000h base + 10h offset + (32d × i), where i=0d to 1d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W		HWACLKSEL														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### Clocking\_CLnKCGHWACSR field descriptions

Field	Description
0 -	This field is reserved. Reserved
1–4 HWACLKSEL	Hardware accelerator clock select. Selects the clock source for peripheral hardware accelerators. See <a href="#">Table 4-6</a> above.  CLKCG1HWACSR[HWACLKSEL] reflects the values programmed in RCW[HWA_CGA_M1_CLK_SEL] and CLKCG2HWACSR[HWACLKSEL] reflects the values programmed in RCW[HWA_CGA_M2_CLK_SEL].  CLKCG1HWACSR (CGA_M1): 0000 Reserved 0001 Reserved 0010 CGA PLL1 divide-by-2 0011 CGA PLL1 divide-by-3 0100 Reserved 0101 Reserved 0110 CGA PLL2 divide-by-2 0111 CGA PLL2 divide-by-3  CLKCG2HWACSR (CGA_M2): 0000 Reserved

Table continues on the next page...

**Clocking\_CLnKCGHWACSR field descriptions (continued)**

Field	Description
	0001 CGA PLL2 divide-by-1 0010 Reserved 0011 CGA PLL2 divide-by-3 0100-1111 Reserved
5–31 -	This field is reserved. Reserved

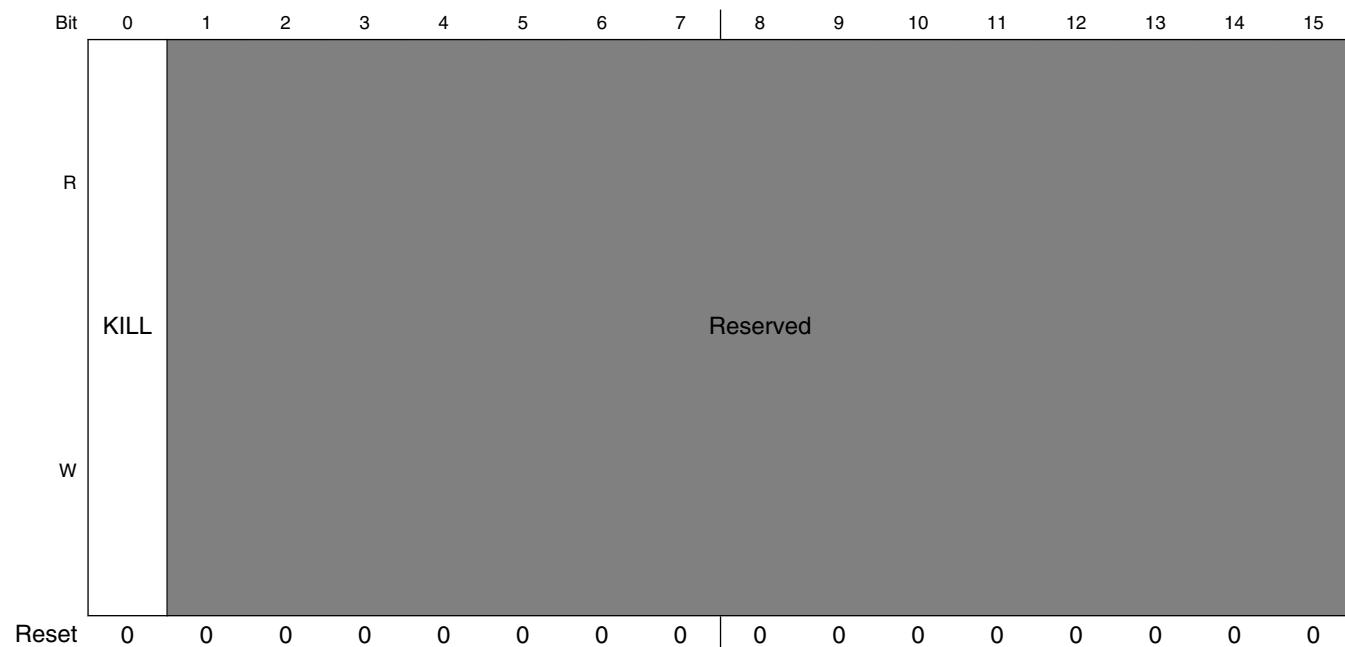
**4.3.3 PLL cluster n general status register (Clocking\_PLLCnGSR)**

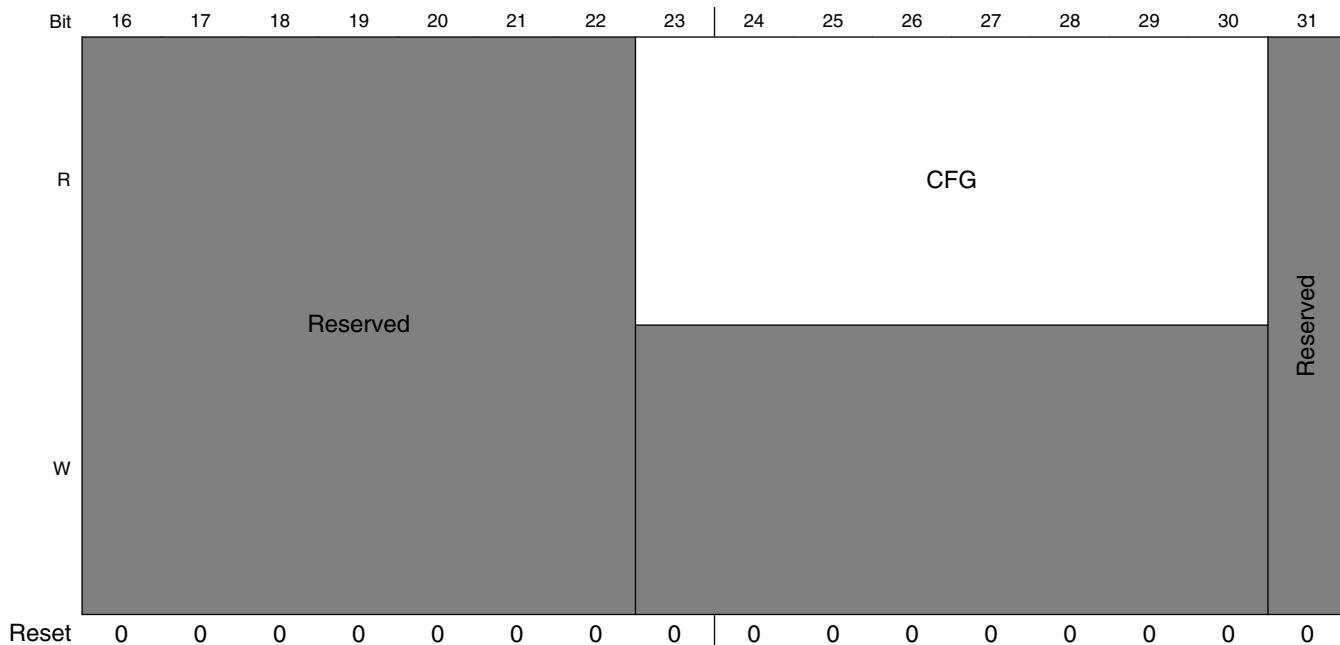
The PLLCnGSR registers provide information regarding the cluster PLL configuration.

**Table 4-7. PLL assignments to PLLCnGSR registers**

Register	Cluster Group	PLL
PLLC1GSR	CGA	PLL1
PLLC2GSR	CGA	PLL2

Address: 1EE\_1000h base + 800h offset + (32d × i), where i=0d to 1d





### Clocking\_PLLCnGSR field descriptions

Field	Description
0 KILL	Writing a 1 to this bit disables the PLL. If PLLnGSR[CFG] indicates 0b00000, the PLL is bypassed and this bit (KILL) is set. 0 PLL is active. 1 PLL is disabled.
1–22 -	This field is reserved. Reserved
23–30 CFG	Reflects the current PLL multiplier configuration. Indicates the frequency for this PLL. PLLC1GSR[CFG] reflects the values programmed in RCW[CGA_PLL1_RAT] and PLLC2GSR[CFG] reflects the values programmed in RCW[CGA_PLL2_CFG].  Defined settings are from 00_0101 (5:1) through 10_1000 (40:1). All other encodings are reserved.  <b>NOTE:</b> Not all ratios are supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
31 -	This field is reserved. Reserved

#### 4.3.4 Platform clock domain control/status register (Clocking\_CLKPCSR)

The CLKPCSR register selects which signal to observe on the CLK\_OUT pad.

Address: 1EE\_1000h base + A00h offset = 1EE\_1A00h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	Reserved								CLKOEN							
W																
R	CLKOSEL					CLKODIV		Reserved								
W																

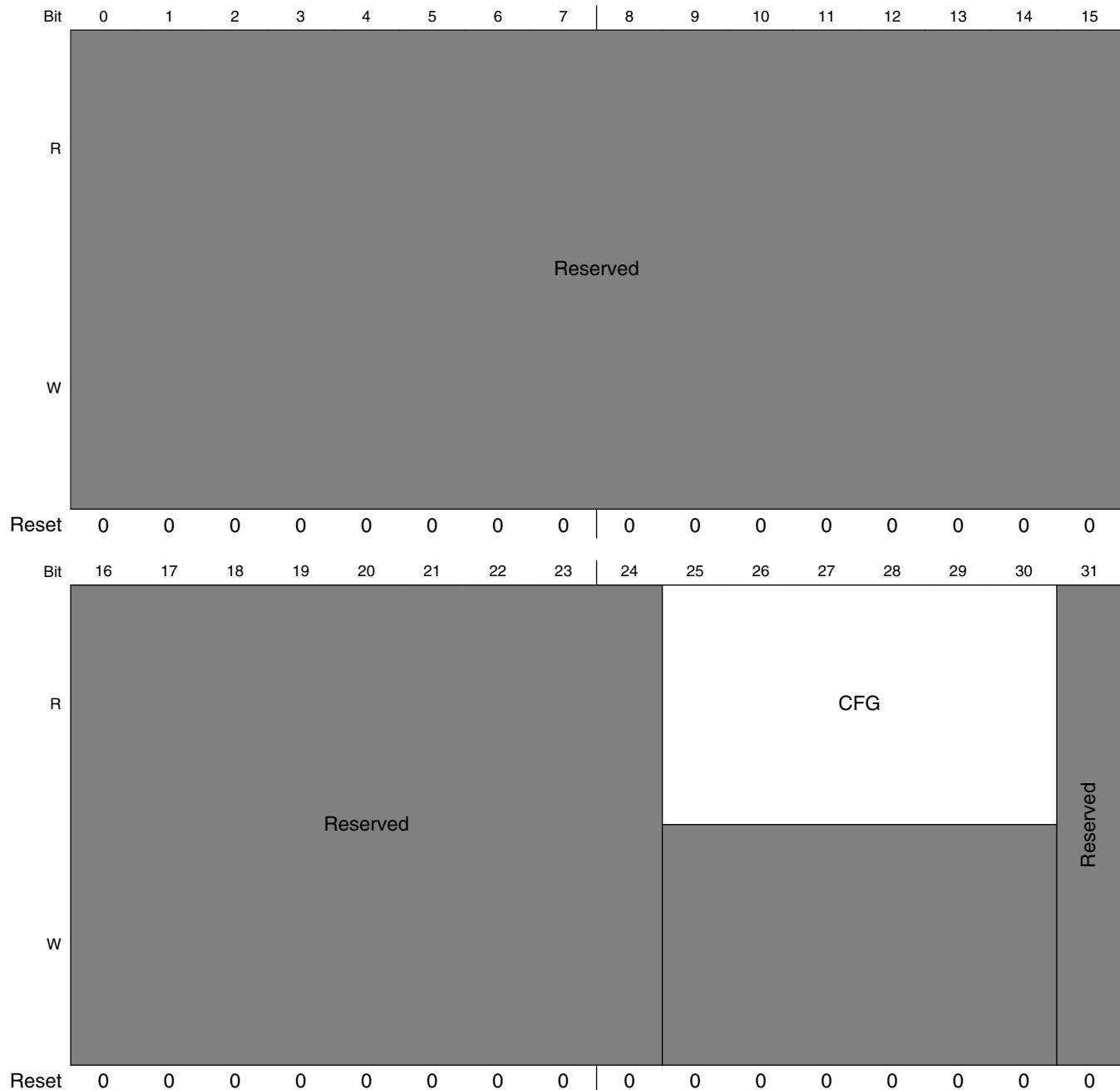
#### Clocking\_CLKPCSR field descriptions

Field	Description
0–14 -	This field is reserved. Reserved
15 CLKOEN	CLK_OUT pad enable 0 Release CLK_OUT pad to high impedance 1 Enable CLK_OUT pad
16–20 CLKOSEL	Selects core related clock signal for observation on CLK_OUT pad 11011 Platform Clock /2 11110 Platform feedback clock 11111 SYSCLK All others Reserved
21–22 CLKODIV	Selects division setting for clock-out mux output to reduce the frequency of the signal to a more observable/measurable range. 00 No division (divide-by-1) 01 Divide-by-2 10 Divide-by-4 11 Divide-by-8
23–31 -	This field is reserved. Reserved

### 4.3.5 Platform PLL general status register (Clocking\_PLLPGSR)

The PLLPGSR register provides information regarding the PLL's configuration.

Address: 1EE\_1000h base + C00h offset = 1EE\_1C00h



**Clocking\_PLLPGSR field descriptions**

Field	Description
0–24 -	This field is reserved. Reserved
25–30 CFG	Reflects the current PLL multiplier configuration. Indicates the frequency for this PLL. Reflects the values programmed in RCW[SYS_PLL_RAT]. Defined settings are from 00_0111 (7:1) through 01_0000 (16:1). All other encodings are reserved. <b>NOTE:</b> Not all ratios are supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
31 -	This field is reserved. Reserved

### 4.3.6 DDR PLL general status register (Clocking\_PLLDGSR)

The PLLDGSR register provides information regarding the DDR PLL configuration.

Address: 1EE\_1000h base + C20h offset = 1EE\_1C20h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
KILL																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
Reserved																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**Clocking\_PLLDGSR field descriptions**

Field	Description
0 KILL	Writing a 1 to this bit disables the PLL. If PLLnGSR[CFG] indicates 0b00000, the PLL is bypassed and this bit (KILL) is set. 0 PLL is active. 1 PLL is disabled.
1–22 -	This field is reserved. Reserved
23–30 CFG	Reflects the current PLL multiplier configuration. Indicates the frequency for this PLL. Reflects the values programmed in RCW[MEM_PLL_RAT].
31 -	This field is reserved. Reserved

## 4.4 Functional description

This section describes the various ways to reset the chip, the POR sequence, the hard reset sequence, the POR configurations, the reset configuration word (RCW), and the clocking on the device.

### 4.4.1 Power-on reset sequence

Assertion of the external PORESET\_B signal initiates the power-on reset flow.

See the chip data sheet for more information regarding power sequencing and PORESET\_B input requirements.

After the negation of PORESET\_B, the reset control logic begins cycling the device through its full reset and RCW configuration process. The chip asserts HRESET\_B throughout the power-on reset process, including RCW configuration. Reset and RCW configuration time varies depending on the configuration source and SYSCLK frequency. Initially, the RCW source POR configuration inputs are sampled to determine the configuration source. Next, the device begins loading the RCW data. The system PLL begins to lock according to the clock ratio/mode values communicated in the RCW data. Once the PLL locks and the RCW data is loaded, HRESET\_B is released by the device and the clocking unit begins distributing PLL outputs throughout the device. Pre-boot initialization is then optionally performed, and the cores are permitted to boot.

The detailed POR sequence for the device is as follows:

1. The external logic drives cfg\_eng\_use0 to a valid state. An internal multiplexer selects between differential and single ended clock correspondingly.

2. The external system logic asserts PORESET\_B and power is applied to comply with the chip's data sheet .
3. PORESET\_B asserted causes all registers to be initialized to their default states and most I/O drivers to be released to high impedance (some clock, clock enables, and system control signals are active).

### **NOTE**

The common on-chip processor (COP) requires the ability to independently assert PORESET\_B and TRST\_B to fully control the processor. If a JTAG/COP port is used, follow the JTAG/COP interface connection recommendations given in the chip's data sheet . If the JTAG interface and COP header are not being used, it is recommended that TRST\_B be tied to PORESET\_B so that TRST\_B is asserted when PORESET\_B is asserted, ensuring that the JTAG scan chain is initialized during the power-on reset flow. See the JTAG configuration signals section in the chip's data sheet for more information.

4. The system applies a toggling SYSCLK signal and stable POR configuration inputs. At this point, SYSCLK is propagated throughout the device; the platform PLL is running in bypass mode.
5. The device begins driving HRESET\_B asserted after sampling the assertion of PORESET\_B.
6. External system logic negates PORESET\_B after its required hold time and after POR configuration inputs have been valid for their required setup times.
7. The device samples the RCW source POR configuration inputs (cfg\_rcw\_src[0:n]) on deassertion of PORESET\_B to determine the RCW source. Note that the POR configuration inputs are sampled only on a PORESET\_B.
8. The device initiates and completes reset of the rest of the platform logic. Note that this platform reset step is the point where the device hard reset process (HRESET\_B) begins if an external device asserts HRESET\_B (assuming the device is not already sequencing through the power-on reset process).
9. Some of the I/O drivers are enabled; specifically, any signals required by the interface specified as the source of RCW data in cfg\_rcw\_src[0:n]. All of the DDR I/Os become enabled at this point (though MCKE, MCK, MODT are enabled from the beginning). The ASLEEP signal is also enabled at this point.
10. If the IFC's NAND Flash interface is configured as the RCW source, the reset block instructs the IFC to load a boot block from Flash into the internal buffer RAM of the IFC. Once complete, the reset block proceeds to instruct the Pre-Boot Loader to begin reading in RCW data. Note that if the IFC NAND Flash interface reports an

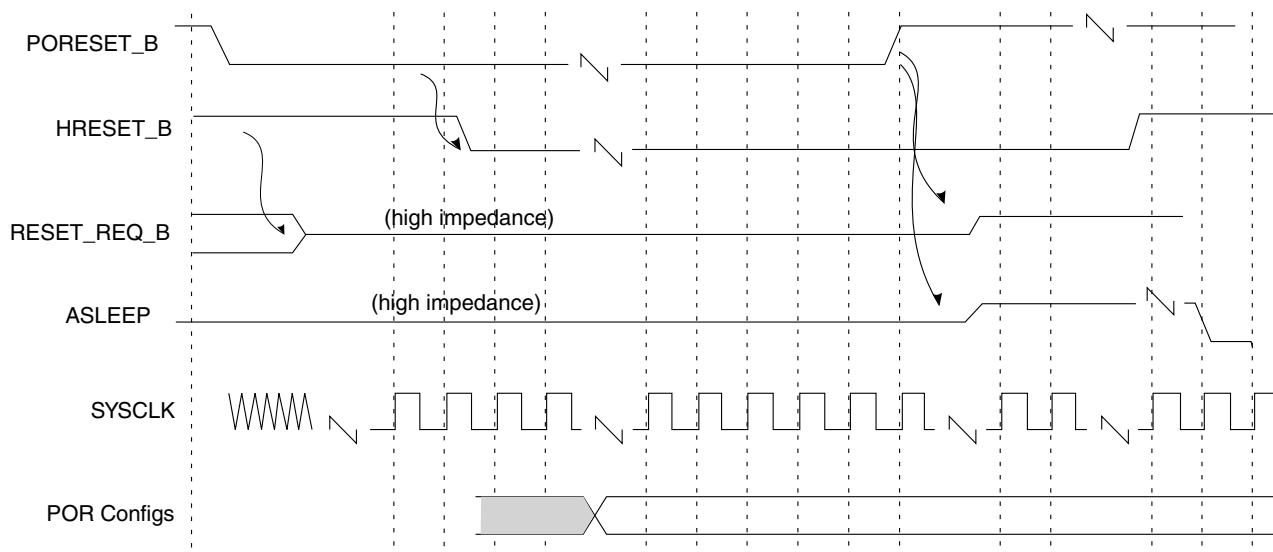
- ECC error, the device reset sequence is halted indefinitely, waiting for another PORESET\_B or hard reset.
11. The pre-boot loader (PBL) starts loading the RCW data from the interface specified by cfg\_rcw\_src[0:n] configuration inputs and stores that 64 bytes of data to the RCWSR registers within the device configuration block. Loading time varies and depends on the source of the RCW. Note that if a hard-coded RCW option is used, this PBL RCW loading process is effectively bypassed and the device is automatically configured according to the specific RCW field encodings pre-assigned for the given hard-coded RCW option (see [Table 4-14](#)) for more information .
  12. The PLLs begin to lock.
  13. The sequence then waits for the PBL RCW process to be completed (loading of all 512 bits). If the PBL reports an error during its process of loading the RCW data, the device reset sequence is halted indefinitely, waiting for another PORESET\_B or hard reset.
  14. The platform clock tree is then switched over and is driven by the output of the platform PLL.
  15. The device stops driving HRESET\_B at this point. All other I/O drivers are enabled at this point.
  16. If the IFC's NAND Flash interface is:
    - configured as the pre-boot initialization source
 OR
    - the boot device target AND not fused as secure boot
 AND
    - the IFC's NAND Flash interface was NOT previously used as the RCW source, then the reset block informs the IFC to load a boot block from Flash into the internal buffer RAM of the IFC. Once complete, the IFC signals back to the reset block, and the reset block can proceed. Note that if the IFC reports an ECC error, the device reset sequence is halted indefinitely, waiting for a hard reset or PORESET\_B.
  17. The PBL performs pre-boot initialization, if enabled by RCW, by reading data from either the eSDHC, QuadSPI, or IFC interface and writing to CCSR space or local memory space (OCRAM1 or OCRAM2, DDR). If the PBL reports an error during its pre-boot initialization process, the device reset sequence is halted indefinitely, waiting for a hard reset or PORESET\_B.
  18. Any external device optionally driving HRESET\_B negates it if not done earlier. If other external devices do not release HRESET\_B, the device reset sequence stalls at this point.

19. System ready state. The peripheral interfaces are released to accept external requests, and the boot vector fetches by the cores are allowed to proceed unless processor booting is further held off by the boot release register (BRR) in the device configuration module. The ASLEEP signal negates synchronized to a rising edge of SYSCLK, indicating the ready state of the system. After reaching this system ready state, the ASLEEP signal transitions to the asserted state when the device enters sleep mode.

### NOTE

After completing reset, software should check the SerDesx\_PLLnRSTCTL[RST\_DONE] field to make sure that each active SerDes PLL on the device has locked. Transactions or packet data cannot be transferred through the targeted lane(s) of the SerDes interface if the PLL associated with the lane(s) does not lock properly. Note that a SerDes PLL will not lock if the corresponding reference clock is not provided.

Figure below shows a timing diagram of the POR sequence.



**Figure 4-1. Power-on reset sequence**

### 4.4.2 Hard reset sequence

The hard reset sequence is initiated by the external system asserting HRESET\_B. Upon sampling the HRESET\_B asserted, the device then begins driving HRESET itself throughout the hard reset state.

Reset and RCW configuration time varies subject to the configuration source and SYSCLK frequency. The reset configuration input signals are not sampled by a hard reset (only a power-on reset), so the device immediately begins loading RCW data and configures the device as explained in [Reset configuration word \(RCW\)](#). After the configuration sequence completes, the device releases the HRESET signal and exits the hard reset state. After negation is detected, a 16-cycle period is taken before testing for the presence of an external reset. The hard reset sequence begins with reset of the device at step 8 in [Power-on reset sequence](#).

#### 4.4.3 Core soft reset

The following steps need to be performed by the software for the core soft reset:

1. Write 1 to [SCFG\\_CORESRENCR\[CORESREN\]](#) bit, to enable the soft reset to the corresponding core.
2. Write 1 to [SCFG\\_CORE0\\_SFT\\_RST\[SOFT\\_RESET\]](#),  
[SCFG\\_CORE1\\_SFT\\_RST\[SOFT\\_RESET\]](#),  
[SCFG\\_CORE2\\_SFT\\_RST\[SOFT\\_RESET\]](#), and  
[SCFG\\_CORE3\\_SFT\\_RST\[SOFT\\_RESET\]](#) to enable the soft reset to the core 0, core 1, core 2, and core 3 respectively. Writing to this bit triggers the corresponding interrupt to respective cores (Interrupt ID 196 and Interrupt ID 197). The interrupts need to be configured as edge trigger interrupt.
3. Perform the GIC interrupt mapping:
  - Enable the interrupt.
  - Select edge trigger mode.
  - Route 196 to core 0, 197 to core 1, 200 to core 2, and 201 to core 3.
4. In the core corresponding ISR will be programmed with WFI.
5. Core executes WFI instructions after receiving the interrupt.
6. Once the core executes WFI instruction, COP generates the corresponding core soft reset.

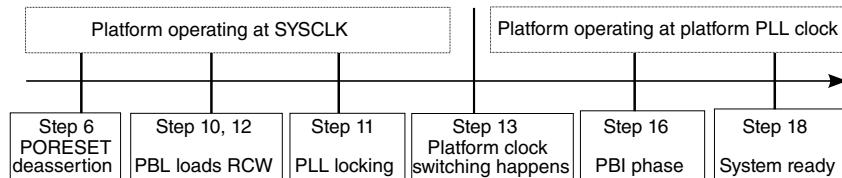
#### 4.4.4 RCW state timing

The following table provides the state timing for different RCW/PBI source.

**Table 4-8. RCW state timing**

RCW/PBI source	RCW loading time (no. of SYSCLK cycles)	RCW source interface frequency (RCW loading phase in terms of SYSCLK ratio)	System PLL locking time (no. of SYSCLK cycles)	RCW source interface frequency (PBI phase in terms of platform clock ratio)	PBI Time(no. of platform clock cycles)	No. of transactions in PBI
NAND-16 (2K page size)	234322	SYSCLK/4	68616	platform clock/8	381814	2 writes of 4 byte each
NOR-16	47260	SYSCLK/4	68616	platform clock/8	9454	2 writes of 4 byte each
SD	811549	SYSCLK/512	68616	platform clock/80	137018	2 writes of 4 byte each
QuadSPI	283242	SYSCLK/256	68616	platform clock/256	19053	2 writes of 4 byte each

The figure below shows the reset timeline diagram in accordance with the power-on reset sequence.



**Figure 4-2. Reset Timeline Diagram**

#### 4.4.5 Power-on reset configuration

Various device functions are initialized by sampling certain signals during power on reset.

The values of all these signals are sampled into registers when PORESET\_B is deasserted. These inputs are to be pulled high or low by external resistors. During PORESET\_B, all other signal drivers connected to these signals must be in the high-impedance state.

All POR configuration signals have internal pull-up resistors so that if the desired setting is high, there is no need for a pull-up resistor on the board. See the chip data sheet for proper pull-down resistor values for POR configuration signals, when necessary.

## Functional description

This section describes the functions and modes configured by the POR configuration signals.

### NOTE

In the following tables, the binary value 0 represents a signal pulled down to GND and a value of 1 represents a signal pulled up to that signal's corresponding V<sub>DD</sub> voltage level, regardless of the sense of the functional signal name.

#### 4.4.5.1 Reset configuration word (RCW) source

The reset configuration word (RCW) source inputs, cfg\_rcw\_src[0:8], are multiplexed on {IFC\_AD[8:15], IFC\_CLE}. These configuration inputs select the source for the RCW data as shown in the following table. The encoded values latched on these signals during POR are accessible in PORSR1[RCW\_SRC], as described in [POR Status Register 1 \(DCFG\\_CCSR\\_PORSR1\)](#).

**Table 4-9. RCW source encodings**

cfg_rcw_src value (Binary)	RCW source
0_0000_xxxx	Reserved
0_0001_xxxx	8-bit NOR Flash  cfg_rcw_src[5]: 0 Address shift "left" (most significant bits are IFC_AD[0:n-1]) 1 Address shift "right" (least significant bits are IFC_AD[0:n-1])  cfg_rcw_src[6:7]: 00 Shift left by 10 to provide 22b addressability OR shift right by appropriate amount (depends on Port Size selected) and provide 22b addressability. 01 Shift left by 7 to provide up to 25b addressability OR shift right by appropriate amount (depends on Port Size selected) and provide up to 25b addressability. 10 Shift left by 4 to provide up to 28b addressability OR shift right by appropriate amount (depends on Port Size selected) and provide up to 28b addressability.  cfg_rcw_src[8]: 0 CS before AVD (address valid supports internal latch based asynchronous NOR devices) 1 AVD before CS (supports simple asynchronous NOR devices)
0_0010_xxxx	16-bit NOR Flash  cfg_rcw_src[5:8] encodings are the same as those described for 0_0001_xxxx (8-bit NOR Flash)
0_0011_000-0_0011_1111	Reserved
0_0100_0000 <sup>1</sup>	SD/MMC (eSDHC)

*Table continues on the next page...*

**Table 4-9. RCW source encodings (continued)**

<b>cfg_rcw_src value (Binary)</b>	<b>RCW source</b>
0_0100_0001-0_0100_0011	Reserved
0_0100_010x <sup>1</sup>	QuadSPI (QuadSPI) <sup>2</sup>
0_0100_0110-0_0100_1001	Reserved
0_0100_1010-0_0111_1111	Reserved
0_100x_xxxx <sup>1</sup>	Reserved
0_1010_0000-0_1111_1111	Reserved
1_0000_00xx	8-bit NAND Flash, 512-byte page, 32 pages/block cfg_rcw_src[7]: 0 Bad Block Indicator in page 0/1 1 Bad Block Indicator in page 0 or last page cfg_rcw_src[8]: 0 ECC disabled 1 4 bits per 520 bytes
1_0000_01xx	8-bit NAND Flash, 2 KB page, 64 pages/block cfg_rcw_src[7:8] encodings are the same as those described for 1_0000_00xx (8-bit NAND Flash, 512-byte page, 32 pages/block)
1_0000_10xx	8-bit NAND Flash, 2 KB page, 128 pages/block cfg_rcw_src[7:8] encodings are the same as those described for 1_0000_00xx (8-bit NAND Flash, 512-byte page, 32 pages/block)
1_0000_11xx	Reserved
1_0001_xxxx	8-bit NAND Flash, 4 KB page, 128 pages/block cfg_rcw_src[5]: 0 Bad Block Indicator in page 0/1 1 Bad Block Indicator in page 0 or last page cfg_rcw_src[6:8]: 000 ECC disabled 001 4 bits per 520 byte sector 010-100 Reserved 101 8 bits per 528 byte sector 110 24 bits per 1 KB sector 111 40 bits per 1 KB sector
1_0010_xxxx	8-bit NAND Flash, 4 KB page, 256 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_0011_xxxx	8-bit NAND Flash, 4 KB page, 512 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_0100_xxxx	8-bit NAND Flash, 8 KB page, 128 pages/block

*Table continues on the next page...*

**Table 4-9. RCW source encodings (continued)**

<b>cfg_rcw_src value (Binary)</b>	<b>RCW source</b>
	cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_0101_xxxx	8-bit NAND Flash, 8 KB page, 256 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_0110_xxxx	8-bit NAND Flash, 8 KB page, 512 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_0111_xxxx	Reserved
1_1000_00xx	16-bit NAND Flash, 512-byte page, 32 pages/block cfg_rcw_src[7:8] encodings are the same as those described for 1_0000_00xx (8-bit NAND Flash, 512-byte page, 32 pages/block)
1_1000_01xx	16-bit NAND Flash, 2 KB page, 64 pages/block cfg_rcw_src[7:8] encodings are the same as those described for 1_0000_00xx (8-bit NAND Flash, 512-byte page, 32 pages/block)
1_1000_10xx	16-bit NAND Flash, 2 KB page, 128 pages/block cfg_rcw_src[7:8] encodings are the same as those described for 1_0000_00xx (8-bit NAND Flash, 512-byte page, 32 pages/block)
1_1000_11xx	Reserved
1_1001_xxxx	16-bit NAND Flash, 4 KB page, 128 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1010_xxxx	16-bit NAND Flash, 4 KB page, 256 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1011_xxxx	16-bit NAND Flash, 4 KB page, 512 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1100_xxxx	16-bit NAND Flash, 8 KB page, 128 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1101_xxxx	16-bit NAND Flash, 8 KB page, 256 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1110_xxxx	16-bit NAND Flash, 8 KB page, 512 pages/block cfg_rcw_src[5:8] encodings are the same as those described for 1_0001_xxxx (8-bit NAND Flash, 4 KB page, 128 pages/block)
1_1111_xxxx	Reserved

1. Not valid as an RCW[IFC\_MODE] encoding
2. Booting from qSPI takes precedence over RCW pin multiplexing which means that QSPI\_A\_DATA[0:3], QSPI\_A\_CS0, QSPI\_A\_CS1, QSPI\_A\_SCK, QSPI\_B\_CS0, QSPI\_B\_CS1, QSPI\_B\_SCK are used regardless of RCW setting.

#### 4.4.5.2 General-purpose input

The general-purpose inputs listed in this table are available for application-specific use.

The encoded values latched on these signals during POR are accessible in GPPORCR1[POR\_CFG\_VEC], as described in [General-Purpose POR Configuration Register \(DCFG\\_CCSR\\_GPPORCR1\)](#).

**Table 4-10. General-purpose input**

Functional signals	Reset configuration name	Value (binary)	General-purpose input
IFC_AD[0:7] Default (1111_1111)	cfg_gpininput[0:7]	all	Application-defined

#### 4.4.5.3 DRAM type select

The DRAM type select input, described in this table, specifies the DDR technology and, thus, voltage ( $GV_{DD}$ ) to be used with the DDR memory controllers.

The encoded value latched on this signal during POR is accessible in PORSR2[DRAM\_TYPE], as described in [POR Status Register 2 \(DCFG\\_CCSR\\_PORSR2\)](#).

**Table 4-11. DRAM type select**

Functional signals	Reset configuration name	Value (binary)	DRAM type
IFC_A21 (Default 1)	cfg_dram_type	0	DDR4 technology (1.2 V)
		1	DDR3L technology (1.35 V)

#### 4.4.5.4 Single oscillator source clock select

The single oscillator source clock select input, described in this table, selects between SYSCLK (single ended) and DIFF\_SYSCLK/DIFF\_SYSCLK\_B (differential) inputs.

**Table 4-12. Single oscillator source clock select**

Functional signals	Reset configuration name	Value (binary)	Options
IFC_WE0_B Default (1)	cfg_eng_use0	0	DIFF_SYSCLK/DIFF_SYSCLK_B (differential)
		1	SYSCLK (single ended)

*Table continues on the next page...*

**Table 4-12. Single oscillator source clock select  
(continued)**

Functional signals	Reset configuration name	Value (binary)	Options
IFC_OE_B	cfg_eng_use1	1	Reserved. Should be set to 1.
IFC_WP_B[0]	cfg_eng_use2	Reserved	Reserved

#### 4.4.5.5 IFC external transceiver enable polarity select

The IFC external transceiver enable polarity select input, described in this table, specifies the polarity of the IFC\_TE output signal to accommodate external transceivers with either active-high or active-low enable inputs.

The encoded value latched on this signal during POR is accessible in PORSR1[IFC\_TE]. See [External transceiver enable \(TE\)](#) for more information.

**Table 4-13. IFC external transceiver enable polarity select**

Functional signals	Reset configuration name	Value (binary)	IFC external transmitter polarity
IFC_TE	cfg_ifc_te	0	IFC drives logic 1 for TE assertion
		1	IFC drives logic 0 for TE assertion

#### 4.4.6 Reset configuration word (RCW)

The chip uses an external memory interface to import a subset of the reset configuration information from a memory device during reset.

Such information is called reset configuration word (RCW) data.

The pre-boot loader (PBL) loads RCW data from a non-volatile memory device interface, as specified by the RCW source configuration inputs (cfg\_rcw\_src[0:8]-see [Reset configuration word \(RCW\) source](#) for more information). See [Pre-Boot Loader](#) for details on the operation of the PBL. Note that this approach does not completely remove the necessity for at least a few power-on reset (POR) configuration signals. As noted, POR config signals are used to control RCW source information in addition to other low-level system configuration.

The logic involved is clocked directly from SYSCLK since RCW importing takes place before on-chip PLLs are configured.

The RCW is 512 bits long in order to contain all necessary configuration information for the chip. RCW data is read from external memory and written to the RCW status registers (see [Reset Control Word Status Register n \(DCFG\\_CCSR\\_RCWSRn\)](#)) contained in the Device Configuration and Pin Control module , after which the device is configured as specified in the RCW. [Required format of data structure used by PBL](#) provides details of the data structure that is required to reside in non-volatile memory.

### NOTE

The chip makes all the required pins available for selected source interface for RCW. Any other multiplexing options on these pins will be overridden.

#### 4.4.6.1 RCW field definitions

This table describes the function of the individual fields of the 512-bit (64-byte) RCW data structure.

### NOTE

Unless noted otherwise, any bit ranges in the table listed as reserved must be populated with 0.

**Table 4-14. RCW Field Descriptions**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
<b>PLL configuration (fields 0-127)</b>			
0-1	SYS_PLL_CFG	System PLL configuration.	Options: 00 For all valid Platform PLL frequencies 01-11 Reserved
2-6	SYS_PLL_RAT	System PLL multiplier/ratio	This field selects the platform clock:SYSCLK ratio. Options: 0_0000 Reserved 0_0011 3:1 0_0100 4:1 0_0101 5:1 0_0110 6:1 0_0111 7:1 0_1000 8:1 0_1001 9:1 0_1010 10:1 0_1011 11:1

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			0_1100 12:1 0_1101 13:1 0_1110 14:1 0_1111 15:1 1_0000 16:1 <b>NOTE:</b> All ratios may not be supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
7	Reserved		
8-9	MEM_PLL_CFG	Memory controller complex PLL configuration (applies to all DDR controllers).	This field configures the memory complex PLLs for the frequency of the reference clock applied. Options: 00 All valid DDR PLL frequencies. 01-11 Reserved
10-15	MEM_PLL_RAT	Memory controller complex PLL multiplier/ratio.	This field configures the DDR PLL:SYSCLK Ratio. Options: 00_0110-Reserved 00_0111-Reserved 00_1000 8:1 00_1001 9:1 00_1010 10:1 00_1011 11:1 00_1100 12:1 00_1101 13:1 00_1110 14:1 00_1111 15:1 01_0000 16:1 01_0001 17:1 01_0010 18:1 01_0011 19:1 01_0100 20:1 01_0101 21:1 01_0110 22:1 01_0111 23:1 01_1000 24:1 01_1001-11_1111-Reserved

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			<b>NOTE:</b> All ratios may not be supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
16-23	Reserved		
24-25	CGA_PLL1_CFG	Cluster group A PLL 1 configuration	Options 00 For Cluster PLL frequencies >= 1 GHz. 01-11 Reserved
26-31	CGA_PLL1_RAT	Cluster group A PLL 1 multiplier/ratio	Options: 00_0101 through 10_1000 5:1 through 40:1 Async All other encodings are reserved. For 1 GHz frequency: The value of this field should be the required multiplication ratio and C1_PLL_SEL should be set to 4'b0000 (CGA_PLL1). For example in order to achieve 1000 MHz core clock frequency with reference clock frequency of 100 MHz, the ratio should be 10(0xA) for locking the CGA PLL1 at 1000 MHz and C1_PLL_SEL=4'b0000 to achieve 1000 MHz core clock frequency. For less than 1 GHz operation: The value of this field should be twice the required core clock frequency and C1_PLL_SEL should be set to 4'b001 (CGA_PLL1/2). For example inorder to achieve 800 MHz core clock frequency with reference clock frequency of 100 MHz, the ratio should be 16(0x10) for locking the CGA PLL1 at 1600 MHz and C1_PLL_SEL=4'b0001 to achieve 800 MHz core clock frequency. <b>NOTE:</b> Not all ratios are supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
32-33	CGA_PLL2_CFG	Cluster group A PLL 2 configuration	Options: 00 00 For Cluster PLL frequencies >= 1 GHz. 01-11 Reserved.
34-39	CGA_PLL2_RAT	Cluster group A PLL 2 multiplier/ratio	Options: 00_0101 5:1 Async 00_0110 6:1 Async 00_0111 7:1 Async 00_1000 8:1 Async 00_1001 9:1 Async 00_1010 10:1 Async 00_1011 11:1 Async 00_1100 12:1 Async

*Table continues on the next page...*

## Functional description

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			00_1101 13:1 Async 00_1110 14:1 Async ..... 10_0111 39:1 Async 10_1000 40:1 Async All other encodings are reserved. <b>NOTE:</b> All ratios may not be supported due to frequency restrictions. Refer to the chip data sheet for the supported frequencies.
40-95	Reserved		
96-99	C1_PLL_SEL	Cluster 1 PLL select.	Options: 0000 CGA_PLL1 /1 0001 CGA_PLL1 /2 0100 CGA_PLL2 /1 0101 CGA_PLL2 /2 All other encodings are reserved. <b>NOTE:</b> All the four cores are in cluster 1 and run at the same frequency.
100-127	Reserved		
<b>SerDes PLL and Protocol configuration (fields 128-183)</b>			
128-143	SRDS_PRTCL_S1	SerDes protocol select SerDes 1	See <a href="#">SerDes protocols</a> for a complete list of the options and the definitions of this encoded field.
144-157	Reserved	Reserved	
158	FM1_MAC_RAT	FM1to MAC1 ratio	Reserved. Must be set as 0'b1.
159	Reserved	Reserved	
160-161	SRDS_PLL_REF_CLK_SEL_S1	SerDes PLL reference clock select - SerDes 1.	This field selects the PLL reference clock frequency for SerDes1 Fields 160: SerDes 1, PLL1 Fields 161: SerDes 1, PLL2 Selection for protocols PCI express operating at 1.25 or 2.5 or 5 GT/s: 0 100 MHz 1 125 MHz Selection for protocols operating at 3 or 6 Gbps 0 100 MHz 1 125 MHz Selection for protocols operating at 3.125 and 10 Gbps: 0 125 MHz

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			1 156.25 MHz (XFI)  <b>NOTE:</b> The higher or lower reference clock frequency depends on the protocol and its operating frequency. See the "Valid reference clocks and PLL configurations for SerDes protocols" section in the SerDes Module chapter for more details.
162-163	Reserved		Set to 0'b0.
164-165	Reserved		
166	HDLC1_MODE	This field selects the HDLC1/TDM pin multiplexing related operating modes including TXD open-drain mode.	HDLC1_MODE option, TXD open-drain mode: 0 - TXD configured in normal functional mode 1 - TXD configured in open-drain mode for either HDLC or TDM.  Note: TXD open-drain mode is not supported at the same time as the TDM Switch Receive Transmit Mode.
167	HDLC2_MODE	This field selects the HDLC2/TDM pin muxing related operating modes including TXD open-drain mode.	HDLC2_MODE option, TXD open-drain mode: 0 - TXD configured in normal functional mode 1 - TXD configured in open-drain mode for either HDLC or TDM.  Note: TXD open-drain mode is not supported at the same time as the TDM Switch Receive Transmit Mode.
168-169	SRDS_PLL_PD_S1	SerDes PLL power down SerDes 1.	This field is used to power down the SerDes 1 PLLs. Field 168 corresponds to SerDes 1, PLL1 and field 169 corresponds to SerDes 1, PLL2.  Option : 0 PLL is not powered down. 1 PLL is powered down.  This field is ignored if the respective SRDS_PRTCL_* field does not use a given SerDes PLL. For more information, refer <a href="#">Disabling unused SerDes modules</a> .
170-175	Reserved		
176-177	SRDS_DIV_PEX	SerDes frequency divider- PCI express	This field controls the frequency of PCI-Express protocols on SerDes lanes that are operating 5/2.5G. Lanes that supporting other frequencies and protocols are unaffected by this field.  Options: 0x Can train up to a max rate of 5G 10 Can train up to a max rate of 2.5G 11 Reserved
178-183	Reserved		
<b>MISC PLL-RELATED (Fields 184-191)</b>			

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
184-185	Reserved, Set to 0'b00.		
186-187	DDR_REFCLK_SEL	DDR reference clock selection	Options: 00 The DDRCLK pin provides the reference clock to the DDR PLL 01 DIFF_SYSCLK/DIFF_SYSCLK_B provides the reference clock to the DDR PLL 10 Reserved 11 Reserved
188	SerDes_REFCLK_SEL	SerDes PLL2 reference clock selection	Selects the reference clock for SerDes PLL2 (used for single oscillator reference clock selection for SerDes) Set to 0'b0.
189	Reserved		
190-191	DDR_FDBK_MULT	DDR PLL feedback path selection and multiplication enabler	Set this value to 10.
<b>Boot configuration (fields 192-223)</b>			
192-195	PBI_SRC	Pre-boot initialization source. The pre-Boot loader fetches address/data pairs from the selected interface for the purpose of pre-boot Initialization of CCSR and/or local memory space.	The following restriction apply: <ul style="list-style-type: none"><li>RCW and pre-boot initialization data must be loaded from the same non-volatile memory device</li></ul> The hard coded RCW source options are not considered their own memory interface for this purpose. Options: 010x QSPI 0110 SD/MMC 1110 IFC (The RCW field IFC_MODE configures the IFC, provided the IFC has not already been configured by the cfg_rcw_src configuration input signals, which have precedence.) All other encodings are reserved.
196-200	Reserved		
201	BOOT_HO	Boot hold off	Options: 0 All cores except core 0 in hold off 1 All cores in hold off
202	SB_EN	Secure boot enable	<b>NOTE:</b> Note that secure boot is enabled if either this RCW field is set or the Intent to Secure fuse value is set. See chapter "Secure Boot and Trust Architecture" for more information. Options: 0 Secure boot is not enabled 1 Secure boot is enabled

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
203-211	IFC_MODE	Integrated Flash controller mode	When PBI_SRC is configured for IFC, this field selects the IFC mode for pre-boot initialization. Note that cfg_rcw_src have precedence over configuring the IFC, see the definition of the PBI_SRC field.  Valid IFC_MODE encodings are a subset of the cfg_rcw_src encodings. See <a href="#">Reset configuration word (RCW) source</a> for the valid encodings for this field.
212-213	Reserved		Set to 00
214-215	Reserved		Set to 00
216-223	Reserved		
<b>Clocking configuration (bits 224-255)</b>			
224-226	HWA_CGA_M1_CLK_SEL	Hardware accelerator block cluster group A, mux 1 clock select.	This controls the async clock frequency provided to FMAN module  Options: 000 - Reserved. 001 Reserved 010 Asynchronous mode - Cluster group A PLL 1 /2 011 Asynchronous mode - Cluster group A PLL 1 /3 100 Reserved 101 Reserved 110 Async mode -- Cluster Group A PLL 2 /2 is clock 111 Async mode -- Cluster Group A PLL 2 /3 is clock
227-229	Reserved		
230-231	DRAM_LAT	DDR latency	Options: 00 6-6-6 or 7-7-7 DRAMs 01 8-8-8, 9-9-9, 10-10-10, 11-11-11, or higher latency DRAMs 10 Reserved 11 5-5-5 DRAMs  <b>NOTE:</b> If DDR latency is not known at the time the RCW is loaded, it is acceptable to conservatively configure this field as 01. This field is used for optimizing the DDR interface. No functional issues result if this field does not match the latency of the actual DRAM's used.
232	DDR_RATE	DDR data rate	Reserved. Must be 0.
233	Reserved		
234	DDR_RSV0	Reserved	Set to 0.
235	Reserved		
236-241	Reserved		Reserved. Must be set to 0.
242	SYS_PLL_SPD	System PLL speed select	Reserved. Must be set to 1.

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
243	MEM_PLL_SPD	Memory controller complex PLL Speed select	Set to 0.
244	CGA_PLL1_SPD	Cluster group A PLL1 speed select	Set to 0.
245	CGA_PLL2_SPD	Cluster Group A PLL2 speed select	Set to 0.
246-255	Reserved		
<b>Memory and high speed I/O configuration (bits 256-287)</b>			
256-263	Reserved		
264-266	HOST_AGT_PEX	Host/agent PEX. Configures Host/Agent mode for all PCI Express Interfaces.	Set to 000 (Host mode). EP mode is not supported.
267-268	Reserved	Set to 0'b0	
269-287	Reserved		
<b>General purpose information (bits 288-319)</b>			
288-295	GP_INFO	General purpose information. This field has no effect on functional logic; it may be used by software.	
296-298	Reserved		
299-319	GP_INFO	General purpose information. This field has no effect on functional logic; it may be used by software.	
<b>Engineering use configuration (bits 320-351)</b>			
320-351	Reserved		
<b>Group A pin configuration (bits 352-383)</b>			
352-353	Reserved		
354-356	UART_EXT	This field configures the functionality of UART pins together with UART_BASE field.	Options: 000 See UART_BASE field definition 001 {UART1_SOUT/GPIO1_15, LPUART1_SOUT, UART1_SIN/GPIO1_17, LPUART1_SIN, LPUART2_SOUT, LPUART1_RTS_B, LPUART2_SIN, LPUART1_CTS_B} 010 {UART1_SOUT/GPIO1_15, LPUART1_SOUT, UART1_SIN/GPIO1_17, LPUART1_SIN, LPUART2_SOUT, LPUART4_SOUT, LPUART2_SIN, LPUART4_SIN} 011 {UART1_SOUT/GPIO1_15, FTM4_CH0, UART1_SIN/GPIO1_17, FTM4_CH1, FTM4_CH2, FTM4_CH3, FTM4_CH4, FTM4_CH5} Settings not shown are reserved For more details, refer <a href="#">UART, GPIO, FTM, and LPUART signal multiplexing</a>
357-359	IRQ_EXT	This field configures the functionality of IRQ[3:11] pins together with IRQ_BASE field.	Options: 000 See IRQ_BASE field definition

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			<p>001 {TDMB_TSYNC, TDMA_TXD/TDMA_RXD, TDMA_RSYNC, TDMA_TXD/TDMA_RXD_EXC, TDMA_TSYNC, TDMB_TXD/TDMB_RXD, TDMB_RSYNC, TDMB_TXD/TDMB_RXD_EXC, GPIO1_31 }</p> <p>010 {UC3_RTSB_TXEN, UC1_RXD7, UC1_CTSB_RXDV, UC1_TXD7, UC1_RTSB_TXEN, UC3_RXD7, UC3_CTSB_RXDV, UC3_TXD7, GPIO1_31}</p> <p>011 {FTM3_CH7, FTM3_CH0, FTM3_CH1, FTM3_CH2, FTM3_CH3, FTM3_CH4, FTM3_CH5, FTM3_CH6, GPIO1_31}</p> <p>Settings not shown are reserved.</p> <p>For more details, refer <a href="#">External IRQ, QE, and GPIO1 signal multiplexing</a></p>
360-362	SPI_EXT	This field configures functionality of SPI pins together with SPI_BASE field.	<p>Options:</p> <p>000 See SPI_BASE field definition.</p> <p>001 {SDHC_CLK_SYNC_OUT, SDHC_CLK_SYNC_IN, Reserved, SDHC_VS, SPI_CS_B[1]/GPIO2_1/SDHC_DAT5/SDHC_CMD_DIR, SPI_CS_B[2]/GPIO2_2/SDHC_DAT6/SDHC_DAT0_DIR, SPI_CS_B[3]/GPIO2_3/SDHC_DAT7/SDHC_DAT123_DIR}</p> <p>010 {SPI_MOSI, SPI_MISO, Reserved, SPI_CS_B[0]/GPIO2_0/SDHC_DAT4, SPI_CS_B[1]/GPIO2_1/SDHC_DAT5/SDHC_CMD_DIR, SPI_CS_B[2]/GPIO2_2/SDHC_DAT6/SDHC_DAT0_DIR, SPI_CS_B[3]/GPIO2_3/SDHC_DAT7/SDHC_DAT123_DIR}</p> <p>Settings not shown are reserved.</p> <p>For more details, refer <a href="#">SPI, eSDHC, USB and GPIO2 signal multiplexing</a></p>
363-365	SDHC_EXT	This field configures the functionality of the SDHC pins together with SDHC_BASE field	<p>Options:</p> <p>000 See SDHC_BASE field definition</p> <p>001 {LPUART3_SOUT, LPUART3_SIN, LPUART2 RTS_B, LPUART2 CTS_B, LPUART3 RTS_B, LPUART3 CTS_B}</p> <p>010 {LPUART3_SOUT, LPUART3_SIN, LPUART5_SOUT, LPUART5_SIN, LPUART6_SOUT, LPUART6_SIN}</p> <p>011 {FTM4_CH6, FTM4_CH7, FTM4_FAULT, FTM4_EXTCLK, FTM4_QD_PHA, FTM4_QD_PHB}</p> <p>Settings not shown are reserved.</p> <p>For more details, refer <a href="#">eSDHC, GPIO2, and GPIO4 signal multiplexing</a></p>

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
366-368	UART_BASE	This field configures functionality of UART pins.	Options: 000 {GPIO1_15, GPIO1_17, GPIO1_19, GPIO1_21, GPIO1_16, GPIO1_18, GPIO1_20, GPIO1_22} 011 {UART1_SOUT, UART1_SIN, GPIO1[19], GPIO1[21], GPIO1[16], GPIO1[18], GPIO1_20], GPIO1_22} } 100 {UART1_SOUT, UART1_SIN, UART1_RTS_B, UART1_CTS_B, GPIO1_16], GPIO1_18, GPIO1_20, GPIO1_22} 101 {UART1_SOUT, UART1_SIN, GPIO1_19, GPIO1_21], UART2_SOUT, UART2_SIN, GPIO1_20], GPIO1_22} 110 {UART1_SOUT, UART1_SIN, UART1_RTS_B, UART1_CTS_B, UART2_SOUT, UART2_SIN, UART2_RTS_B, UART2_CTS_B} 111 {UART1_SOUT, UART1_SIN, UART3_SOUT, UART3_SIN, UART2_SOUT, UART2_SIN, UART4_SOUT, UART4_SIN} <b>NOTE:</b> Note that UART_EXT field must be set to all 0's for UART_BASE to take effect.
369	ASLEEP	This field configures functionality of ASLEEP pin.	Options: 0 ASLEEP 1 GPIO1_13
370	RTC	This field configures functionality of RTC pin.	Options: 0 RTC 1 GPIO1_14
371	SDHC_BASE	This field configures functionality of SDHC pins: {{SDHC_CMD, SDHC_DAT[0:3], SDHC_CLK}	Options: 0 {SDHC_CMD, SDHC_DAT[0:3], SDHC_CLK} 1 GPIO2[4:9] <b>NOTE:</b> If cfg_rcw_src selects SD/MMC as the RCW source, the SDHC pins are driven with SDHC functionality regardless of the setting of this field.
372	IRQ_OUT	This field configures functionality of the IRQ_OUT pin.	Reserved. Set to 1.
373-381	IRQ_BASE	This field configures functionality of IRQ[3:11] pins. The corresponding GPIOs for these pins are GPIO1[23:31].	Options for each bit: 0 IRQ 1 GPIO
382-383	SPI_BASE	This field configures functionality of the SPI_CS_B[0:3] pins.	Options: 00 SPI_CS_B[0:3], SPI_MOSI, SPI_MISO, SPI_CLK 01 SDHC_DAT[4:7] for 8-bit MMC card support

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			10 GPIO2[0:3] 11 {Reserved, SDHC_CMD_DIR, SDHC_DAT0_DIR, SDHC_DAT123_DIR}
<b>Group B pin configuration (bits 384-415)</b>			
384-386	IFC_GRP_A_EXT	This field configures the functionality of Group A of the IFC pins together with IFC_GRP_A_BASE field.	<p>Options:</p> <ul style="list-style-type: none"> <li>000 See IFC_GRP_A_BASE field definition</li> <li>001 {QSPI_A_DATA[3]}</li> <li>010 {FTM5_CH0, FTM5_CH1, FTM5_EXTCLK}</li> <li>100 {IFC_CS_B[4], IFC_CS_B[5], IFC_CS_B[6]}</li> </ul> <p>Settings not shown are reserved.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• If IFC_CS_B[5] is used, then IFC_RB_B[3] cannot be used and vice-versa</li> <li>• If QuadSPI is selected for cfg_rcw_src then this field must be set to 001.</li> </ul>
387-392	Reserved		
393-395	IFC_GRP_D_EXT	This field configures the functionality of Group D of the IFC pins, together with IFC_GRP_D_BASE field.	<p>Options:</p> <ul style="list-style-type: none"> <li>000 See IFC_GRP_D_BASE field definition</li> <li>001 {QSPI_B_DATA[0], QSPI_B_DATA[1], QSPI_B_DATA[2]}</li> <li>010 {FTM6_CH0, FTM6_CH1, FTM6_EXTCLK}</li> </ul> <p>Settings not shown are reserved</p>
396-398	IFC_GRP_E1_EXT	This field configures functionality of Group E1 of the IFC pins, together with IFC_GRP_E1_BASE field.	<p>Options:</p> <ul style="list-style-type: none"> <li>000 See IFC_GRP_E1_BASE field definition</li> <li>001 {IFC_CS_B[1]/GPIO2_10, IFC_CS_B[2]/GPIO2_11, QSPI_B_DATA[3]}</li> <li>010 {FTM7_CH0, FTM7_CH1, FTM7_EXTCLK}</li> </ul> <p>Settings not shown are reserved.</p>
399-401	IFC_GRP_F_EXT	This field configures functionality of Group F of the IFC pins.	<p>Options:</p> <ul style="list-style-type: none"> <li>000 {IFC_A[16:24]/IFC_WP_B[1:3]}</li> <li>001 {QSPI_A_CS0, QSPI_A_CS1, QSPI_A_SCK, QSPI_B_CS0, QSPI_B_CS1, QSPI_B_SCK, QSPI_A_DATA[0], QSPI_A_DATA[1], QSPI_A_DATA[2]}</li> </ul> <p>Settings not shown are reserved.</p> <p><b>NOTE:</b> If QuadSPI is selected for cfg_rcw_src then this field must be set to 001.</p>
402-404	IFC_GRP_G_EXT	This field configures functionality of Group G of the IFC pins.	Settings not shown are reserved.

*Table continues on the next page...*

## Functional description

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
405	IFC_GRP_E1_BASE	Together with IFC_GRP_E1_EXT field, this field configures functionality of Group E1 of the IFC pins.	Options: 0 IFC_CS_B[1:3] 1 {GPIO2[10:12]}
406	Reserved		
407	IFC_GRP_D_BASE	Together with IFC_GRP_D_EXT field, This field configures functionality of Group D of the IFC pins.	Options: 0 {IFC_PAR[0:1]/IFC_PERR_B} 1 {GPIO2[13:15]}
408	Reserved		
409-411	Reserved		
412-413	IFC_GRP_A_BASE	This field configures functionality of Group A of the IFC pins.	Options: 00 {IFC_A[25:27]} 01 {GPIO2[25:27]} 10 {IFC_RB_B[2:3], Reserved}  Whenever IFC is selected for less than 28 bits (25-bits or 22-bits) using cfg_rcw_src, this field should be set such that IFC_AD is not selected. When IFC is selected as 25-b or 22-b addressing,RCW[IFC_GRP_A_BASE] should not be set to 00.
414	Reserved		
415	IFC_A_22_24	This field configures functionality of the IFC pins IFC_A[22:24] for 16b data bus pinouts which pinout 25b or 28b of addressability.	Options: 0 {IFC_A[22:24]} 1 {IFC_WP_B[1:3]}  Whenever IFC is selected for 22-b functionality using cfg_rcw_src, this field should be chosen such that IFC_AD is not selected. And, When IFC is selected as 22-b addressing, RCW[IFC_A_22_24] should not be set to 0.
<b>SoC-Specific configuration (bits 416-447)</b>			
416-418	EC1	Selects the functionality assigned to the EC1 pins.	Options: 000 RGMII1 001 GPIO3 010 Reserved 011 Reserved 100 Reserved 101 FTM1 110 Reserved 111 Reserved
419-421	EC2	Selects the functionality assigned to the EC2 pins.	Options: 000 RGMII2

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			001 GPIO3, GPIO3[19:23] 010 IEEE 1588 011 Reserved 100 Reserved 101 FTM2 110 Reserved 111 Reserved  When configured for IEEE1588, the EC2 pins that are not available for IEEE1588 are configured for GPIO.
422-424	Reserved		
425	EM1 (MDC_MDIO)	Configures functionality of the EM1 MDC_MDIO pins.	Options: 0 MDC/MDIO (EM1) 1 GPIO_3
426	EM2 (MDC_MDIO)	Configures functionality of the EM2 MDC_MDIO pins.	Options: 0 MDC/MDIO (EM2) 1 GPIO_4
427	EMI2_DMODE	This field selects the EMI 2, Ethernet management interface MDIO pin multiplexing related operating modes.	MDIO Open-Drain mode: Recommended setting for this field is 0'b1. Options: 0 MDIO configured in normal functional mode 1 MDIO configured in Open-Drain mode.
428	EMI2_CMODE	This field selects the EMI 2-Ethernet management interface MDC pin multiplexing related operating modes.	MDC Open-Drain mode: Recommended setting for this field is 0'b1. Options: 0 MDC configured in normal functional mode 1 MDC configured in Open-Drain mode.
429	USB_DRVVBUS	This field configures the functionality of the USB_DRVVBUS pin.	Options: 0 USB_DRVVBUS 1 GPIO4_29 <b>NOTE:</b> Refer <a href="#">USB DRVVBUS Control Register (SCFG_USBDRVVBUS_SELCR)</a> for USB_DRV_VBUS to USB Controller mapping.
430	USB_PWRFAULT	This field configures the functionality of the USB_PWRFAULT signal.	Options: 0 USB_PWRFAULT 1 GPIO4_30 <b>NOTE:</b> Refer <a href="#">USB PWRFAULTControl Register (SCFG_USBPWRFAULT_SELCR)</a> for

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			USB_PWRFAULT to USB Controller mapping.
431-432	Reserved		
433-434	TVDD_VSEL	Configures voltage of the TVDD IO domain.	Options: 00 1.2 V or 1.8 V 01 2.5 V 10 Reserved 11 Auto-voltage selection enabled. This is used only for hard-coded RCW values.
435-436	DVDD_VSEL	Configures voltage of the DVDD IO domain.	Options: 00 1.8V 01 Reserved 10 3.3V 11 Auto-voltage selection enabled. This is used only for hard-coded RCW values.
437	QE_CLK_OVERRIDE	Configure the selection of QE clocks on IIC2_SCL.	Options: 0 Select as per IIC2_EXT(RCW[445-447]). Options for IIC2_EXT:RCW[445-447] available are: 101 - CLK9, CLK10 110 - BRGO2,BRGO3 1 Toggle the I2C_SCL functionality. Options for IIC2_EXT:RCW[445-447] available are: 101 - BRGO2, CLK10 110 - CLK9,BRGO3 <b>NOTE:</b> This overrides only for CLK9 Vs BRGO2 on IIC2_SCL pin. If QE functionality is not selected then this pin should be set to 0.
438	EMI1_DMODE	This field selects the EMI 1, Ethernet management interface MDIO pin multiplexing related operating modes.	MDIO Open-Drain mode: Recommended setting for this field is 0'b1. Options: 0 MDIO configured in normal functional mode 1 MDIO configured in Open-Drain mode.
439-440	EVDD_VSEL	Configures voltage of the EVDD IO domain.	Options: 00 1.8V 01 Reserved 10 3.3V 11 Auto-voltage selection enabled. This is used only for hard-coded RCW values or while the voltage transition is happening from 3.3 V to 1.8 V.

*Table continues on the next page...*

**Table 4-14. RCW Field Descriptions (continued)**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
441-443	IIC2_BASE	This field configures functionality of the IIC pins.	Set to 0b000.
444	EMI1_CMODE	This field selects the EMI 1-Ethernet management interface MDC pin multiplexing related operating modes.	MDC Open-Drain mode: Recommended setting for this field is 0'b1. Options: 0 MDC configured in normal functional mode 1 MDC configured in Open-Drain mode.
445-447	IIC2_EXT	Selects between IIC2 base functionality and extension	Options: 000 IIC2_SCL, IIC2_SDA 001 SDHC_CD_B, SDHC_WP 010 GPIO4_2, GPIO4_3 011 FTM3_QD_PHA, FTM3_QD_PHB 100 QE_SI1_STROBE[0], QE_SI1_STROBE[1] 101 CLK9, CLK10 110 BRGO2, BRGO3 Settings not shown are reserved.
448-471	Reserved		
472-481	SYSCLK_FREQ	SYSCLK frequency	This field is used for proper hardware configuration of the Arm generic timer and by software to determine the frequency of SYSCLK. The value in this field is multiplied by 166.667 KHz. It is used to provide information to determine the frequency of operation of the Arm Generic Timer. The allowable range depends on the range of SYSCLK frequencies supported.  The frequency for Arm generic timer is SYSCLK/4.  Example values(in decimal, not hexadecimal): 10'd384 - 64 MHz 10'd400 - 66.667 MHz 10'd500 - 83.3 MHz 10'd600 - 100 MHz
482-493	Reserved		
494-508	Reserved		Set to 0'b0.
509-511	HWA_CGA_M2_CLK_SEL	Hardware accelerator block cluster group A mux 2 clock select.	This field allows for a platform accelerator block's (or multiple blocks') frequency to be maximized (using async clock) by leveraging cluster group A mux 2.  This controls the async clock frequency provided to eSDHC and QuadSPI.  Options: 001 Async mode, Cluster Group A PLL 2 /1 is clock 011 Async mode, Cluster Group A PLL 2 /3 is clock

**Table 4-14. RCW Field Descriptions**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
			<p>Setting not shown are reserved.</p> <p>When eSDHC used in MMC4.5 in SDR mode:</p> <ul style="list-style-type: none"> <li>• CGA2PLL should be set for 1200 MHz</li> <li>• HWA_CGA_M2_CLK_SEL should be set for 3'b001</li> <li>• /6 occurs local to the eSDHC</li> <li>• Net result is a 200 MHz clock</li> </ul> <p>When eSDHC used in MMC4.2 or MMC4.5 DDR mode:</p> <ul style="list-style-type: none"> <li>• Settings above must be chosen to generate a clock that is 300 MHz. This can be accomplished using 900 MHz / 3</li> <li>• /6 occurs local to the eSDHC</li> <li>• Net result is a 50 MHz clock</li> </ul>

#### 4.4.6.2 Hard-coded RCW options

If any of the hard-coded RCW options are used, the PBL RCW loading process is bypassed and the device is automatically configured according to the specific RCW field encodings that are pre-assigned for the given hard-coded RCW option. The hard-coded RCW options can be useful if the board has no valid RCW present or if the customer is unable to load in the RCW from a non-volatile memory. Using a hard-coded RCW option allows the part to be initialized so that the desired RCW can be programmed or restored.

##### **NOTE**

Hard-coded RCW option is not recommended as a feature to be used for functional bring-up (DDR and SerDes functionality). It can be used to bring-up the board with blank flash where flash needs to be programmed for RCW information through JTAG interface.

The hard-coded RCW option should be selected based on the system SYSCLK and DDRCLK frequency combination.

##### **NOTE**

If a hard-coded RCW option is used, the user should not enable the core and the DDR controllers. The user may enable the core and DDR controllers after a valid RCW is restored.

**Table 4-15. Hard-Coded RCW Options**

<b>cfg_rcw_src[0:8] Encoding</b>	<b>Comments</b>	
0_1001_1010 (0x9A)	SYSCLK	DDRCLK
	66 MHz	100 MHz
0_1001_1110 (0x9E), 0_1001_1111 (0x9F)	SYSCLK	DDRCLK
	100 MHz	100 MHz

**4.4.6.3 RCW settings for hard-coded Options****Table 4-16. RCW Settings for hard-coded RCW options**

<b>RCW Field</b>	<b>cfg_rcw_src[0:8] = 0_1001_1010 (0x9A)</b>	<b>cfg_rcw_src[0:8] = 0_1001_1110 (0x9E)</b>	<b>cfg_rcw_src[0:8] = 0_1001_1111 (0x9F)</b>		
<b>PLL CONFIGURATION (BITS 0-127)</b>					
SYS_PLL_CFG	0's				
SYS_PLL_RAT (must be an even ratio)	4:1				
MEM_PLL_CFG	0's				
MEM_PLL_RAT	13:1				
CGA_PLL1_CFG	0's				
CGA_PLL1_RAT	18:1	12:1			
CGA_PLL2_CFG	0's				
CGA_PLL2_RAT	15:1	10:1			
C1_PLL_SEL	CGA_PLL1 / 1				
<b>SerDes PLL AND PROTOCOL CONFIGURATION (BITS 128-183)</b>					
SRDS_PRTCL_S1	SerDes 1 set to a value of 0x0000 (All lanes used)				
FM1_MAC_RAT	1'b1				
SRDS_PLL_REF_CLK_SEL_S1	0'b00 (SerDes 1 Refclk: PLL1 = 100 MHz, PLL2 = 100 MHz)				
HDLC1_MODE	2'b00				
HDLC2_MODE	2'b00				
SRDS_PLL_PD_S1	PLL1 = not powered down, PLL2 = powered down				
SRDS_DIV_PEX	2'b01 (5G)				
<b>MISC PLL-RELATED (BITS 184-191)</b>					
DDR_REFCLK_SEL	2'b00	2'b01			
SerDes_REFCLK_SEL	1'b1				
DDR_FDBK_MULT	2'b10				
<b>BOOT CONFIGURATION (BITS 192-223)</b>					

*Table continues on the next page...*

## Functional description

**Table 4-16. RCW Settings for hard-coded RCW options (continued)**

RCW Field	cfg_rcw_src[0:8] = 0_1001_1010 (0x9A)	cfg_rcw_src[0:8] = 0_1001_1110 (0x9E)	cfg_rcw_src[0:8] = 0_1001_1111 (0x9F)		
PBI_SRC	Disabled				
BOOT_HO	All cores other than core 0				
SB_EN	Secure boot disabled				
IFC_MODE	0-don't care				
<b>CLOCKING CONFIGURATION (BITS 224-255)</b>					
HWA_CGA_M1_CLK_SEL	Cluster Group A PLL 2 /2				
DRAM_LAT	Set to 2'b01.				
DDR_RATE	Divide by 2				
DP_DIV	2'b00				
OCN_DIV	2'b00				
SYS_PLL_SPD	1'b1	1'b0			
MEM_PLL_SPD	1'b0				
CGA_PLL1_SPD	1'b0				
CGA_PLL2_SPD	1'b0				
<b>MEMORY AND HIGH SPEED I/O CONFIGURATION (BITS 256-287)</b>					
RIO_DEVICE_ID	0'b0				
RIO_SYS_SIZE	0'b0				
HOST_AGT_PEX	All in Agent Mode				
<b>GENERAL PURPOSE INFORMATION (BITS 288-319)</b>					
GP_INFO	0x0				
<b>ENGINEERING USE CONFIGURATION (BITS 320-351)</b>					
ENG_USE	All Zeros				
<b>CHASSIS GROUP A PIN CONFIGURATION (BITS 352-383)</b>					
UART_EXT	All Zeros (configured by UART_BASE)				
IRQ_EXT	All Zeros (configured by IRQ_BASE)				
SPI_EXT	All Zeros (configured by SPI_BASE)				
UART_BASE <sup>1</sup>	All Zeros (all GPIOs)				
ASLEEP	Zero (Asleep)				
RTC	Zero (RTC)				
SDHC_BASE	Zero (SD/MMC)				
IRQ_OUT	All Zeros (EVT_B[9])				
IRQ_BASE	All Zeros (All IRQs)				
SPI_BASE	All Zeros (All Chip Selects of SPI)				
<b>CHASSIS GROUP B PIN CONFIGURATION (BITS 384-415)</b>					
IFC_GRP_A_EXT	All Zeros (configured by IFC_GRP_A_BASE)				
IFC_GRP_B_EXT	All Zeros (configured by IFC_GRP_B_BASE)				
IFC_GRP_C_EXT	All Zeros (configured by IFC_GRP_C_BASE)				
IFC_GRP_D_EXT	All Zeros (configured by IFC_GRP_D_BASE)				

*Table continues on the next page...*

**Table 4-16. RCW Settings for hard-coded RCW options (continued)**

RCW Field	<code>cfg_rcw_src[0:8] = 0_1001_1010 (0x9A)</code>	<code>cfg_rcw_src[0:8] = 0_1001_1110 (0x9E)</code>	<code>cfg_rcw_src[0:8] = 0_1001_1111 (0x9F)</code>
IFC_GRP_E1_EXT	All Zeros (configured by IFC_GRP_E1_BASE)		
IFC_GRP_E2_EXT	Zero (configured by IFC_GRP_E2_BASE)		
IFC_GRP_F_EXT	All Zeros (configured by IFC_GRP_F_BASE)		
IFC_GRP_G_EXT	All Zeros (configured by IFC_GRP_G_BASE)		
IFC_GRP_E1_BASE	Zero (IFC Chip Selects 1-3)		
IFC_GRP_E2_BASE	Zero (IFC Chip Selects 4-7)		
IFC_GRP_D_BASE	Zero (IFC Parity)		
IFC_GRP_C_BASE	Zero (IFC Address bits 26-31)		
IFC_GRP_B_BASE	All Zeros (IFC Address/Data bits 28-31)		
IFC_GRP_A_BASE	All Zeros (IFC Address/Data bits 25-27)		
<b>SoC SPECIFIC CONFIGURATION (BITS 416-447)</b>			
EC1	3'b001 (GPIO)		
EC2	3'b001 (GPIO)		
EM1 (MDC/MDIO)	1'b1 (GPIO)		
EM2 (MDC/MDIO)	1'b1 (GPIO)		
EMI2_DMODE	1'b0 (Normal functional mode)		
EMI2_CMODE	1'b0 (Normal functional mode)		
USB_DRVVBUS	1'b1 (GPIO)		
USB_PWRFAULT	1'b1 (GPIO)		
TVDD_VSEL	2'b11 (Auto)		
DVDD_VSEL	2'b11 (Auto)		
QE_CLK_OVERRIDE	1'b0		
EMI1_DMODE	1'b0 (Normal functional mode)		
EVDD_VSEL	2'b11 (Auto)		
IIC2_BASE	3'b000		
EMI1_CMODE	1'b0 (Normal functional mode)		
IIC2_EXT	3'b010 (GPIO)		
<b>PLL AND CLOCKING CONFIGURATION EXPANSION (BITS 448-511)</b>			
SYSCLK_FREQ	10'h190	10'h258	
HWA_CGA_M2_CLK_SEL	3'b001 (Cluster Group A PLL 2 divide by 1)		

1. By default UART is not configured, so boot code need to override the RCW for UART prompt in hard-coded RCW mode.

## 4.4.7 Clocking

The following sections describe the clocking within the chip.

### 4.4.7.1 Single source clocking

The chip supports the single source clocking options with single, two, and more reference oscillators.

#### 4.4.7.1.1 Single oscillator source reference clock mode

In this mode, single on-board oscillator would provide the single reference clock (100 MHz) to the following PLLs:

- Platform PLL
- Core PLLs
- USB PLL
- DDR PLL

And, require two additional reference clocks for:

- SerDes PLLs

#### **NOTE**

The chip will not complete the reset sequence if SerDes reference clocks are not provided and the PLLs are enabled in the RCW (RCW\_SRDS\_PLL\_PD\_S1).

The reset configuration field identifies whether the SYSCLK (single-ended) or DIFF\_SYSCLK (differential) is selected as the clock input to the chip.

The RCW[DDR\_REFCLK\_SEL] bit is used to select clock input (DIFF\_SYSCLK or DDRCLK) to the DDR PLL.

The following figure shows the system view of single oscillator source clocking. In this figure, the on-board oscillator generates three differential clock outputs. The first differential output is used to provide the clock to system clock associated PLLs and DDR PLL. However, the second and third differential outputs are used to provide clocks to SerDes PLLs.

A multiplexer between system clock and DIFF\_SYSCLK/DIFF\_SYSCLK\_B is used to provide the USB PHY reference clock to the USB PLL. And, multiplexer between DIFF\_SYSCLK/DIFF\_SYSCLK\_B inputs and DDRCLK is used to provide reference clock to the DDR PLL.

The duty cycle reshaper reshapes the 125 MHz ECn\_GTX\_CLK125 which is fed into frame manager for transmission as ECn\_GTX\_CLK.

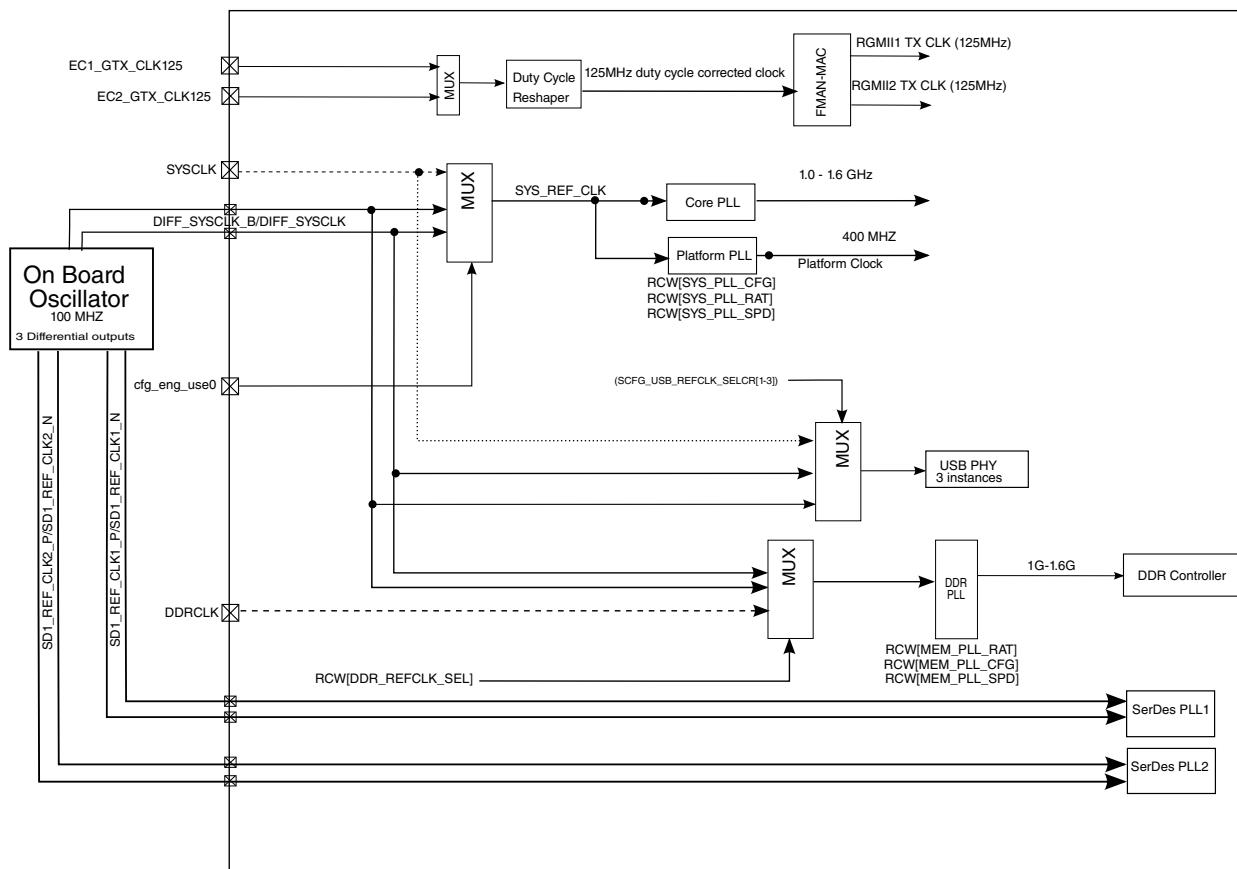


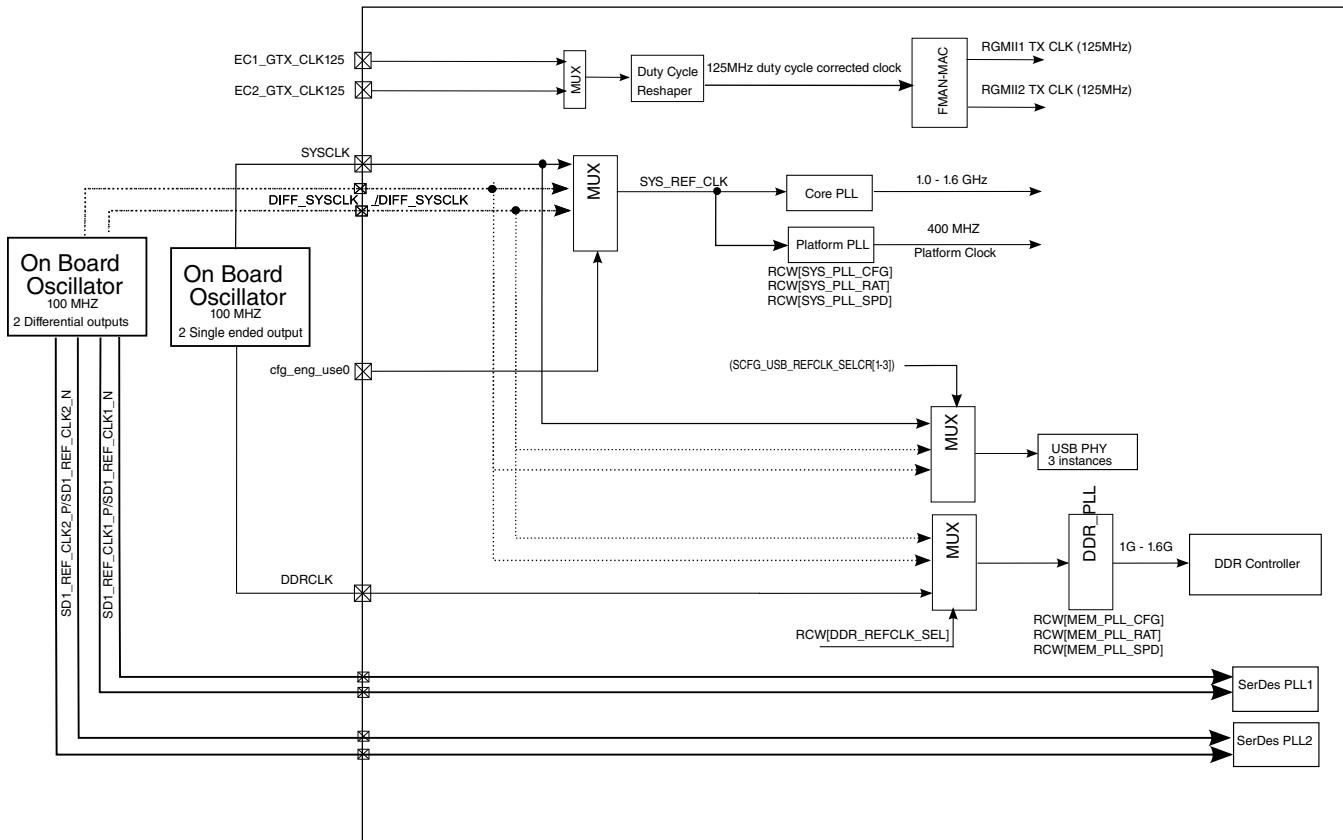
Figure 4-3. Single Oscillator Source Clocking

#### 4.4.7.1.2 Dual reference clock mode

In this configuration, a single on-board oscillator must provide two copies of the single ended reference clock (maximum frequency is 100 MHz), one for the following PLLs:

- Platform PLL, Core PLLs, USB PHY
- DDR PLL

A separate differential clock is provided to the SerDes.



**Figure 4-4. Dual reference clocking**

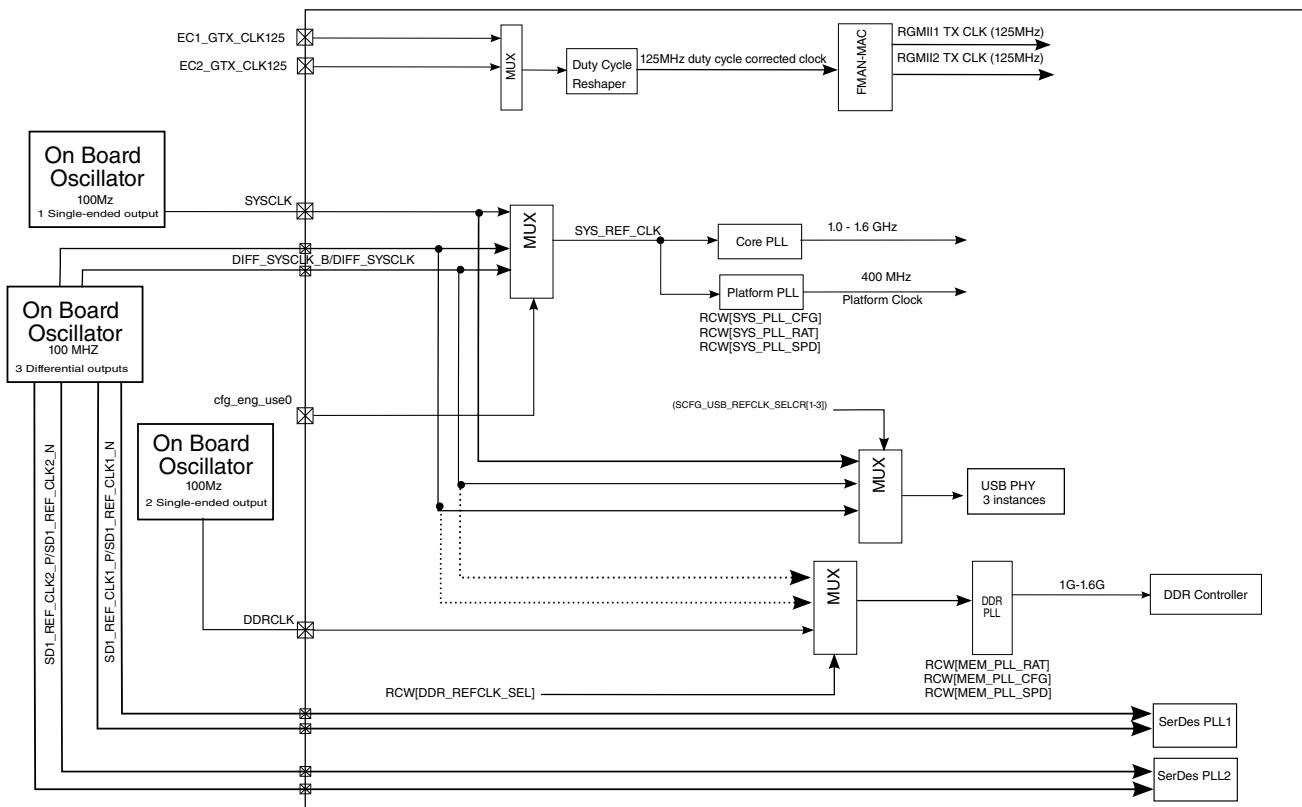
#### 4.4.7.1.3 Multiple reference clock mode

In this configuration, the SYSCLK signal provides the reference clock for the following PLLs:

- Platform PLL
- Core PLL

The DDR PLL is clocked by the DDRCLK signal. This clock is selected through RCW[DDR\_REFCLK\_SEL] and one or two additional reference clocks are provided to the SerDes.

The USB3 PLLs is clocked from either SYSCLK or DIFF\_SYSCLK/DIFF\_SYSCLK\_B. The selection of this clock is through SCFG\_USB\_REFCLK\_SELCR $n$  register.



**Figure 4-5. Multiple reference clocking**

#### 4.4.7.2 IP logic clock distribution and configuration

The chip takes primary clocking input from the external SYSCLK signal. As shown in [Figure 4-6](#), the SYSCLK input (frequency) is multiplied using multiple phase locked loops (PLL) to create a variety of frequencies which can then be passed to a variety of internal logic, including cores and peripheral IP modules.

The DDR PLL is used to provide clocking to the DDR memory controller complexes. The DDR PLL may use the DIFF\_SYSCLK/DIFF\_SYSCLK\_B input clock as a reference to create a unique DDR memory controller complex clock. In this case, the DDR complex operates asynchronously with respect to the platform clock.

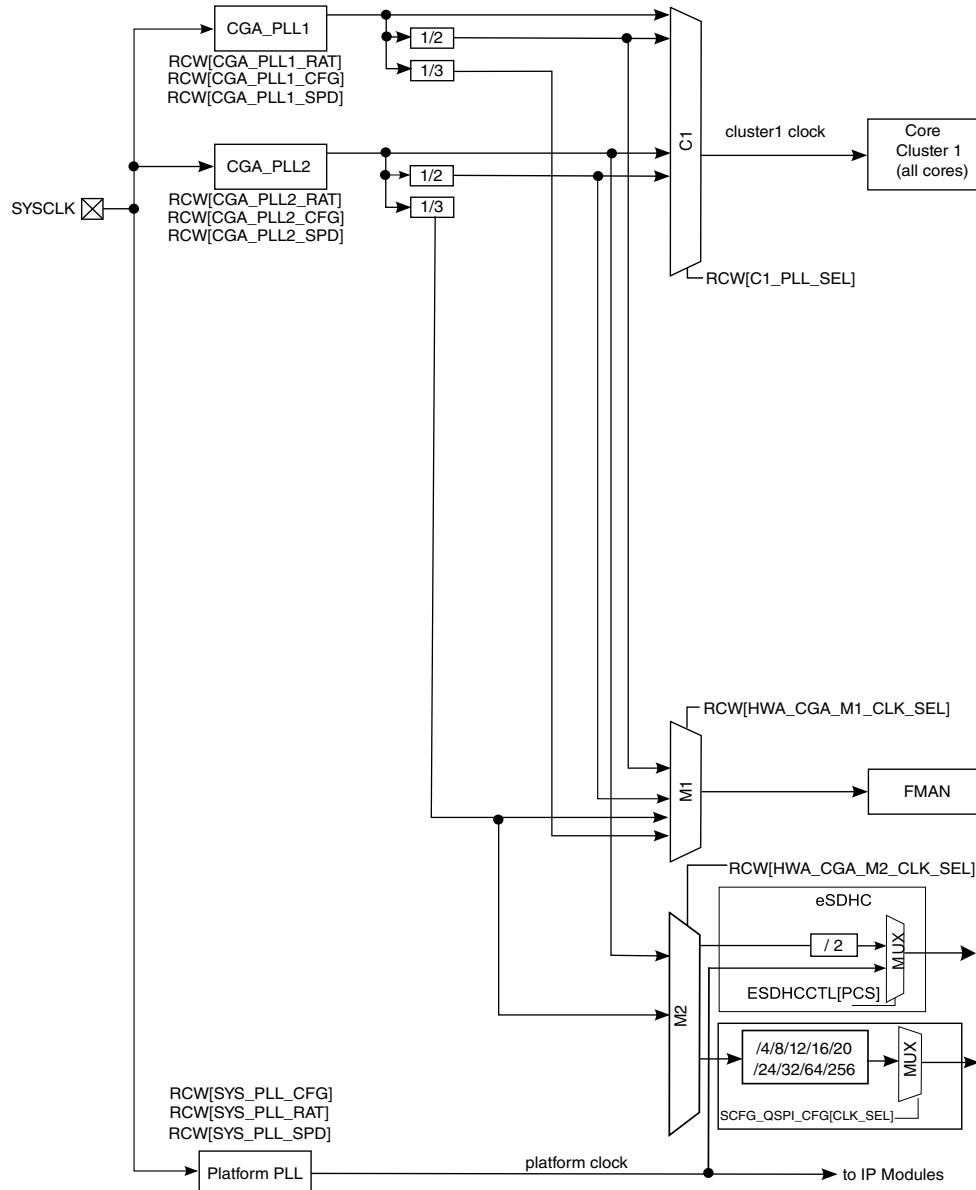
Note that many of the IP modules contain logic allowing software to further modify their external interface clocks within the IP module. See the applicable IP module chapter for details.

The following figures describes internal logic clock distribution along with the means to configure the various ratios and clock sources. Note that the following figures are not intended to reflect external interface clocking. Although sometimes dependent on or derived from logic clocking, a full description of external interface clocking is described

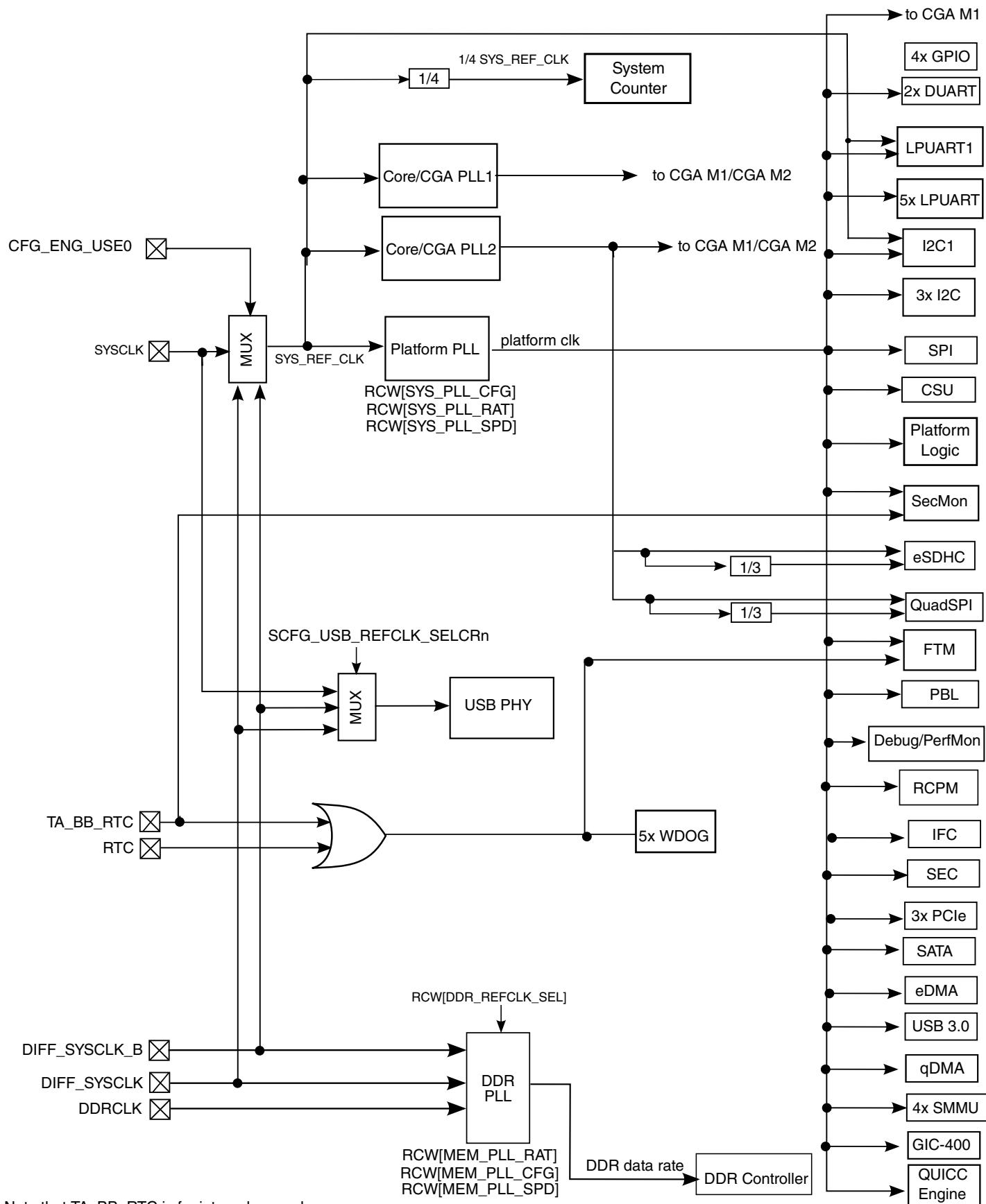
within the applicable IP module chapter of this reference manual. Each of the four SerDes bank external interfaces are clocked by dedicated SerDes reference clock inputs (SD1\_REF\_CLK1/SD1\_REF\_CLK2). (See [Valid reference clocks and PLL configurations for SerDes protocols](#) for details regarding valid combinations of external reference clocks and RCW configurations.)

### NOTE

For any operations above 52 MHz, eSDHC must be clocked by PLL source by setting eSDHCCTL[PCS] to 1.



**Figure 4-6. Clock subsystem block diagram - cluster group A**



Note that TA\_BB\_RTC is for internal use only.

**Figure 4-7. Clock subsystem block diagram - IP modules**  
QorIQ LS1043A Reference Manual, Rev. 6, 07/2020

#### 4.4.7.3 CLK\_OUT configuration

The CLK\_OUT signal can be configured to offer external hardware one of a variety of internal clock signals for debug or diagnostic purposes.

(See [Platform clock domain control/status register \(Clocking\\_CLKPCSR\)](#) for details.)

# Chapter 5

## Interrupt Assignments

### 5.1 Introduction

This chapter describes GIC interrupt assignments for the chip. These are the on-chip interrupt sources from peripheral logic within the integrated device.

### 5.2 Internal interrupt sources

The implementation of the GIC handles up to 256 interrupt requests to the Arm core. The following table shows the assignments of the internal interrupt sources.

**Table 5-1. Interrupt assignments**

Internal Interrupt Number	Interrupt Source	Comments
0-31	Arm internal interrupts	These are software generated interrupts for communication among cores and are triggered through GIC-400 registers. See <a href="#">Arm generic interrupt controller (GIC-400)</a> for more information.
32	FTM5	All FTM interrupts are ORed together.
33	FTM6	See <a href="#">FTM Interrupts</a> for more information.
34-63	Reserved	
64	Reserved	
65	Thermal monitor unit alarm	See <a href="#">TMU interrupt enable register (TIER)</a> for more information.
66	Thermal monitor unit critical alarm	
67-70	Reserved	
71	qDMA INT0	Virtualized qDMA
72	qDMA INT1	See <a href="#">Command Queue Interrupts</a> for more information.
73	qDMA INT2	
74	qDMA INT3	

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
75	IFC	All IFC interrupts are ORed together and connected to this interrupt.  See <a href="#">Common Event and Error Interrupt Enable register (IFC_CM_EVTER_INTR_EN)</a> for more information.
76	Frame manager (FMan)	See DPAA reference manual.
77	FMAN/QMAN/BMAN error	
78	MDIO management interrupt 1 (1 G)	
79	MDIO management interrupt 2 (10 G)(XFI)	
80	LPUART1	All LPUART interrupts are ORed together.
81	LPUART2	See <a href="#">Interrupts and status flags</a> for more information.
82	LPUART3	
83	LPUART4	
84	LPUART5	
85	LPUART6	
86	DUART1	See <a href="#">Interrupt control logic</a> for more information.
87	DUART2	
88	I2C1	All I2C interrupts are ORed together.
89	I2C2	See <a href="#">I2C Bus Status Register (I2C_IBSR)</a> for more information.
90	I2C3	
91	I2C4	
92	USB1	See USBSTS register in <a href="#">Table 36-3</a> for more information.
93	USB2	
94	eSDHC (SD/MMC)	See <a href="#">Interrupt status register (IRQSTAT)</a> for more information.
95	USB3	See USBSTS register in <a href="#">Table 36-3</a> for more information.
96	SPI1	See <a href="#">Interrupts/DMA requests</a> for more information.
97	Reserved	
98	GPIO1	All GPIO interrupts are ORed together.
99	GPIO2	See <a href="#">GPIO interrupt event register (GPIER)</a> for more information.
100	GPIO3	
101	SATA 3.0	
102	EPU	All EPU interrupts are ORed together and connected to this interrupt.
103	SEC job queue 1	For more information, refer Security Reference Manual.
104	SEC job queue 2	
105	SEC job queue 3	
106	SEC job queue 4	
107	SEC global	

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
108	Platform control (Miscellaneous system control module (MSCM))	All Platform Control interrupts are ORed together and connected to this interrupt. See <a href="#">Miscellaneous System Control Module (MSCM)</a> for more information.
109	uQE (QE interrupt + QE critical + QE error)	
110	SecMon secure	See SecMon_HP Security Violation Status Register (HPSVSR) for more information.
111	Secmon non-secure	See SecMon_HP Status Register (HPSR) for more information.
112	CSU	
113	WDOG3	See <a href="#">Interrupt event</a> for more information.
114	WDOG4	
115	WDOG1	
116	WDOG2	
117	WDOG5	
118	FTM1	All FTM interrupts are ORed together. See <a href="#">FTM Interrupts</a> for more information.
119	FTM2	
120	FTM3	
121	FTM4	
122	Reserved	
123	FTM7	All FTM interrupts are ORed together. See <a href="#">FTM Interrupts</a> for more information.
124	FTM8	
125	TZASC	All TZASC interrupts are ORed together and connected to this interrupt. See <a href="#">TrustZone Address Space Controller</a> for more information.
126	A53 core 2 CTI IRQ	Core 0 and core 1 cross trigger interrupt See <a href="#">Arm® Cortex®-A53 core</a> for more information.
127	A53 core 2 PMU IRQ	
128	A53 core 3 CTI IRQ	
129	A53 core 3 PMU IRQ	
130	Reserved	
131	QuadSPI	All QuadSPI interrupts are ORed together and connected to this interrupt. See <a href="#">Interrupt Signals</a> for more information.
132-134	Reserved	
135	eDMA	
136	A53 core 0 CTI IRQ	Core 0 and core 1 cross trigger interrupt See <a href="#">Arm® Cortex®-A53 core</a> for more information.
137	A53 core 1 CTI IRQ	
138	A53 core 0 PMU IRQ	
139	A53 core 1 PMU IRQ	

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
140	A53 AXI Error / A53 Int Err	It represents the combined interrupt for L1 D-cache ECC error (for all cores) and L2 cache errors. See <a href="#">Arm® Cortex®-A53 core</a> for more information.
141	CCI400 ERRORIRQ / CCI EVNTCNTOVERFLOW	CCI-400 error interrupt CCI-400 transaction bus error due to any transaction error in CCI-400 interconnect and connected to nERRORIRQ of CCI-400. See <a href="#">The CCI-400 module as implemented on the chip</a> for more information.
142	PEX1 INT (INTA, INTB, INTC, or INTD)	See <a href="#">PCI Express Interrupt Pin Register (Interrupt_Pin_Register)</a> for more information.
143	PEX MSI1 INT2	See <a href="#">PCI Express MSI implementation</a> for more information.
144	PEX MSI1 INT3	
145	PEX MSI1 INT4	
146-147	Reserved	
148	PEX MSI1 INT1	See <a href="#">PCI Express MSI implementation</a> for more information.
149	PEX1 PME	Refer PM_PME messages in the Power Management chapter of PCI Express™ Base Specification, Revision 3.0 for more information.
150	PEX1 CFG err interrupt	
151	Reserved	
152	PEX2 INT (INTA, INTB, INTC, or INTD)	See <a href="#">PCI Express Interrupt Pin Register (Interrupt_Pin_Register)</a> for more information.
153	PEX MSI2 INT2	See <a href="#">PCI Express MSI implementation</a> for more information.
154	PEX MSI2 INT3	
155	PEX MSI2 INT4	
156-157	Reserved	
158	PEX MSI2 INT1	See <a href="#">PCI Express MSI implementation</a> for more information.
159	PEX2 PME	Refer PM_PME messages in the Power Management chapter of PCI Express™ Base Specification, Revision 3.0 for more information.
160	PEX2 CFG err interrupt	
161-162	Reserved	
163	IRQ0	IRQ0-11 are external IO signals connected to interrupt controller. The polarity of these IRQ's are programmable in <a href="#">Interrupt Polarity Register (SCFG_INTPCR)</a> .  The external IO signals(IRQ0-11) are directly connected to interrupt lines and hence the status of these interrupts are available in the corresponding GIC-400 registers.
164	IRQ1	
165	IRQ2	

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
166	GPIO4	All GPIO interrupts are ORed together. See <a href="#">GPIO interrupt event register (GPIER)</a> for more information.
167	IRQ3	IRQ0-11 are external IO signals connected to interrupt controller. The polarity of these IRQ's are programmable in <a href="#">Interrupt Polarity Register (SCFG_INTPCR)</a> .
168	IRQ4	
169	IRQ5	The external IO signals(IRQ0-11) are directly connected to interrupt lines and hence the status of these interrupts are available in the corresponding GIC-400 registers.
170	qDMA	
171-173	Reserved	
174	SMMU-500 (TCU) non-secure	This is a single non-secure interrupt from SMMU and ORed of the following interrupts: <ul style="list-style-type: none"> <li>• Configuration fault interrupt</li> <li>• Global fault interrupt</li> <li>• Performance interrupt</li> <li>• Context interrupt</li> </ul> See <a href="#">System memory management unit (MMU-500)</a> for more information.
175	SMMU-500 (TCU) secure	
176	DDR controller (error)	<a href="#">Memory error interrupt enable (ERR_INT_EN)</a> for more information.
177	IRQ6	IRQ0-11 are external IO signals connected to interrupt controller. The polarity of these IRQ's are programmable in <a href="#">Interrupt Polarity Register (SCFG_INTPCR)</a> .
178	IRQ7	
179	IRQ8	The external IO signals(IRQ0-11) are directly connected to interrupt lines and hence the status of these interrupts are available in the corresponding GIC-400 registers.
180	Reserved	
181	IRQ9	IRQ0-11 are external IO signals connected to interrupt controller. The polarity of these IRQ's are programmable in <a href="#">Interrupt Polarity Register (SCFG_INTPCR)</a> .
182	IRQ10	
183	IRQ11	The external IO signals(IRQ0-11) are directly connected to interrupt lines and hence the status of these interrupts are available in the corresponding GIC-400 registers.
184	Reserved	
185	qDMA error interrupt	LS1043A assigns individual interrupts (71-74) to each core.
186	PEX3 INT (INTA, INTB, INTC or INTD)	See <a href="#">PCI Express Interrupt Pin Register (Interrupt_Pin_Register)</a> for more information.

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
187	PEX MSI3 INT2	See <a href="#">PCI Express MSI implementation</a> for more information.
188	PEX MSI3 INT3	
189	PEX MSI3 INT4	
190-191	Reserved	
192	PEX MSI3 INT1	See <a href="#">PCI Express MSI implementation</a> for more information.
193	PEX3 PME	Refer PM_PME messages in the Power Management chapter of PCI Express™ Base Specification, Revision 3.0 for more information.
194	PEX3 CFG err interrupt	
195	Reserved	
196	Soft reset core 0	Edge triggered only. This interrupt facilitates core 0 soft reset sequence and should be configured as edge interrupt. See <a href="#">Core soft reset</a> for more information.
197	Soft reset core 1	Edge triggered only. This interrupt facilitates core 1 soft reset sequence and should be configured as edge interrupt. See <a href="#">Core soft reset</a> for more information.
198-199	Reserved	
200	Soft reset core 2	Edge triggered only. This interrupt facilitates core 2 soft reset sequence and should be configured as edge interrupt. See <a href="#">Core soft reset</a> for more information.
201	Soft reset core 3	Edge triggered only. This interrupt facilitates core 3 soft reset sequence and should be configured as edge interrupt. See <a href="#">Core soft reset</a> for more information.
202-203	Reserved	
204	Queue manager portal 0	See DPAA reference manual.
205	Buffer manager portal 0	
206	Queue manager portal 1	
207	Buffer manager portal 1	
208	Queue manager portal 2	
209	Buffer manager portal 2	
210	Queue manager portal 3	
211	Buffer manager portal 3	
212	Queue manager portal 4	
213	Buffer manager portal 4	
214	Queue manager portal 5	
215	Buffer manager portal 5	
216	Queue manager portal 6	

*Table continues on the next page...*

**Table 5-1. Interrupt assignments (continued)**

Internal Interrupt Number	Interrupt Source	Comments
217	Buffer manager portal 6	
218	Queue manager portal 7	
219	Buffer manager portal 7	
220	Queue manager portal 8	
221	Buffer manager portal 8	
222	Queue manager portal 9	
223	Buffer manager portal 9	
224	COMMIRQ0	
225	COMMIRQ1	
226	COMMIRQ2	
227	COMMIRQ3	
228	MBEE	Internal RAM Multi-bit ECC Error
229	Virtual CPU interface maintenance interrupt Core 0	
230	Virtual CPU interface maintenance interrupt Core 1	
231	Virtual CPU interface maintenance interrupt Core 2	
232	Virtual CPU interface maintenance interrupt Core 3	
233	Reserved	Edge interrupt
234	PEX 1 Hot Reset	Edge interrupt
235	PEX 1 Link Down	Edge interrupt
236	PEX 1 Link Up	Edge interrupt
237	Reserved	Edge interrupt
238	PEX 2 Hot Reset	Edge interrupt
239	PEX 2 Link Down	Edge interrupt
240	PEX 2 Link Up	Edge interrupt
241	Reserved	Edge interrupt
242	PEX 3 Hot Reset	Edge interrupt
243	PEX 3 Link Down	Edge interrupt
244	PEX 3 Link Up	Edge interrupt
245-255	Reserved	



# Chapter 6

## Arm Modules

### 6.1 Introduction

The chip implements the following Arm modules:

- Arm® Cortex®-A53 core
- Arm generic interrupt controller (GIC-400)
- System memory management unit (MMU-500)
- Cache coherent interconnect (CCI-400)
- Arm CoreLink™ TrustZone address space controller (TZC-380)

This chapter provides a brief overview of the core, interrupt controller, and memory management unit. For more information on these modules, see the Arm documentation that accompanies this reference manual. For the latest released documentation for the possible updates, visit [Arm information center](#).

**Table 6-1. Related resources from Arm**

Resource	IP Revision
Arm® Cortex®-A53 MPCore Processor Technical Reference Manual	r0p4
CoreLink™ GIC-400 Generic Interrupt Controller Technical Reference Manual	r0p1
Arm® CoreLink™ MMU-500 System Memory Management Unit Technical Reference Manual	r2p2

For information on other Arm modules, see the following:

- [Cache Coherent Interconnect \(CCI-400\)](#)
- [Arm CoreLink™ TrustZone Address Space Controller \(TZC-380\)](#)

## 6.2 Arm® Cortex®-A53 core

The Arm® Cortex®-A53 processor is an extremely power efficient Armv8 processor capable of supporting 32-bit and 64-bit code seamlessly. It makes use of a highly efficient 8-stage in-order pipeline balanced with advanced fetch and data access techniques for performance.

The multicore processing provides the ability for any of the four component processors, within a cluster, to shut down when not in use, for instance when the device is in standby mode, to save power. When higher performance is required, every processor is in use to meet the demand while still sharing the workload to keep power consumption as low as possible.

The LS1043A features four high-performance Cortex A53 cores:

- 64 and 32-bit execution states for scalable high performance
- Multiple coherent SMP processor clusters through AMBA® 4 technology
- New instruction set, A64
- In-order pipeline with symmetric dual-issue of most instructions
- 32 KB Instruction Cache, 32 KB Data Cache, 1 MB unified L2 Cache.
- NEON technology - Accelerates multimedia and signal processing algorithms such as video encode/decode, 2D/3D graphics, gaming, audio and speech processing, image processing, telephony, and sound synthesis. Also useful in accelerating floating point code with SIMD execution.
- Hardware-accelerated cryptography - 3x-10x better software encryption performance  
Useful for small granule decrypt/encrypt too small to efficiently offload to HW accelerator
- Floating point unit - Hardware support for floating point operations in half-, single- and double-precision floating point arithmetic. IEE754-2008 enhancements are included.
- TrustZone® Technology - Ensures reliable implementation of security applications ranging from digital rights management to electronic payment.
- Double Precision Floating Point SIMD - Allows SIMD vectorisation to be applied to a much wider set of algorithms (for example scientific / high performance computing (HPC) and supercomputer).
- 64-bit Virtual address reach - Enables virtual memory beyond 4GB 32b limit.  
Important for modern desktop and server software using memory mapped file I/O, sparse addressing.
- Enhanced Cache management - User space cache operations improve dynamic code generation efficiency, data cache zero for fast clear.

- Extensive power-saving features - Hierarchical clock gating, power domains, advanced retention modes
- Hardware virtualization support
- Arranged as a cluster of four cores sharing a single 1 MB L2 cache
- NEON SIMD extensions onboard (per core)
- Single-threaded cores with 32 KB L1 data cache and 32 KB L1 instruction cache
- In-order pipeline with symmetric dual-issue of most instructions.

LS1043A contains one Arm Cortex A53 MPCore cluster, each with four 64-bit Arm A53 v8 cores connected to a shared L2 cache. The following table summarizes the core-cluster sub-system implementation within the chip:

**Table 6-2. Summary: core cluster sub-system implementation**

A53 MPCore parameter	Options	Selection	Comments
Number of cores	1, 2, 3, 4	4	
L1 Icache size	8K, 16K, 32K, 64K	32K	
L1 Dcache size	8K, 16K, 32K, 64K	32K	Applies to all cores
L2 cache	Included, not included	Included	Applies to all cores
L2 cache size	128K, 256K, 512K, 1024K, 2048K	1024K	
L2 data RAM input latency	1 cycle, 2 cycles	1 cycle	
L2 data RAM output latency	2 cycles, 3 cycles	2 cycles	
SCU L2 cache protection	Included, not included	Included	
Advanced SIMD and floating-point extension	Included, not included	Included	Applies to all cores
Cryptography extension	Included, not included	Included	Applies to all cores
CPU cache protection	Included, not included	Included	Applies to all cores
Master memory interface	AMBA 5 CHI, AMBA 4 ACE	AMBA 4 ACE	
Accelerator coherency port	Included, not included	Included	
Debug memory map	v7, v8	v8	

## 6.3 Arm generic interrupt controller (GIC-400)

Generic interrupt controller (GIC) is a centralized resource for supporting and managing interrupts in a system that includes at least one processor. It provides:

- registers for managing interrupt sources, interrupt behavior, and interrupt routing to one or more processors
- support for the following:
  - the Arm architecture security extensions
  - the Arm architecture virtualization extensions

### System memory management unit (MMU-500)

- enabling, disabling, and generating processor interrupts from hardware (peripheral) interrupt sources
- Software-generated interrupts (SGIs) interrupt masking and prioritization uniprocessor and multiprocessor environments

For LS1043A, GIC-400 registers memory-map is aligned to 64 KB.

**Table 6-3. GIC memory allocation**

Start address	End address	Size	Allocation
0x0140_0000	0x0140_FFFF	64K	Reserved
0x0141_0000	0x0141_FFFF	64K	Distributor
0x0142_0000	0x0142_FFFF	64K	CPU interfaces (GICC_CTLR available at 0x142_0000)
0x0143_0000	0x0143_FFFF	64K	CPU interfaces (GICC_DIR available at 0x143_0000)
0x0144_0000	0x0144_FFFF	64K	Virtual interface control block, for the processor that is performing the access
0x0145_0000	0x0145_FFFF	64K	Virtual interface control block, for the processor selected by address bits [11:9]
0x0146_0000	0x0146_FFFF	64K	Virtual CPU interfaces (GICV_CTLR available at 0x146_F000)
0x0147_0000	0x0147_FFFF	64K	Virtual CPU interfaces (GICV_DIR available at address 0x147_0000)

## 6.4 System memory management unit (MMU-500)

The MMU-500 is a system-level Memory Management Unit (MMU) that translates an input address to an output address, by performing one or more translation table walks.

It supports the translation table formats defined by the Arm architecture, and can perform

- Stage 1 translations that translate an input Virtual Address (VA) to an output Physical Address (PA) or Intermediate Physical Address (IPA).
- Stage 2 translations that translate an input IPA to an output PA.
- Combined stage 1 and stage 2 translations that translate an input VA to an output IPA, and then translate that IPA to a PA. The MMU-500 performs a translation table walk for each stage of the translation.
- The purpose of separating the process into 2 stages is to allow the guest OS to control the address translation between VA and what it “thinks” is the PA, while the Hypervisor controls the translation from IPA to PA. This split process allows the Hypervisor to separate the resources of different Virtual Machines (VMs). Address translation is performed both in the cores and also for IO devices, using the System Memory Management Unit (SMMU).

The MMU-500 is a distributed SMMU, which makes use of one central controller (TCU) and up to 32 translation units (TBUs). In this chip there is a single TCU, supporting a total of 4 TBUs. The SMMU supports 8 Stream ID bits and SSD. The security attributes and other security features depends on the SSD.

The SSD is secure state determination. In LS1043A, the SSD table is initialized to contain only two entries 0 (secure transaction) and 1 (non-secure transaction). And, these entries are indexed by the NS\_IN attribute (0 for secure and 1 for non-secure) of the transaction.

Each master has an unique stream ID assigned to it. The StreamID is an identifier attached to each transaction in order to determine the translation context. The SMMU uses a hit/miss mechanism for the following concatenation {tbu number, stream\_id}. This concatenation (and not just the stream id) is then assigned (if exist) to a context bank that determines the translation type and form. Another parameter that affects the translation is the SSD value. Secure transactions can only be subjected to a stage 2 translation, so this value also plays a part in the translation process.

The isolation context identifier (ICID) maps an incoming transaction from IO device to one of the contexts, it maps to StreamID as described in Arm documentations. All the SMMU support 8 ICID bits and SSD index. The address translation depends only on the ICID of the incoming transaction. The security attributes and other security features depends on the SSD index.

Some masters have an unique ICID assigned to it and is configurable through SCFG registers. The ICID is an identifier attached to each transaction in order to determine the translation context. The SSD index and ICID for the masters are identical and share the same register field of ICID registers.

**Table 6-4. ICID connectivity**

IO Device	ICID Connectivity	
	IP Driven	SCFG Register
FMan	ICID output of FMan	not configured through SCFG registers
QMan/BMan	ICID output of QMan/BMan	not configured through SCFG registers
SEC	ICID output of SEC	not configured through SCFG registers
PCI express 1, 2, 3	ICID output of PCI Express	not configured through SCFG registers
qDMA	ICID output of qDMA	not configured through SCFG registers
SATA	-	One register to define 8 bits for ICID. Refer SATA ICID register (SATA_ICID) in chapter "Supplemental Configuration Unit".
USB 1, 2, 3	-	One register to define 8 bits for ICID. Refer USB1 ICID register (USB1_ICID), USB2 ICID Register (USB2_ICID), and USB3 ICID Register (USB3_ICID) in chapter "Supplemental Configuration Unit".

*Table continues on the next page...*

**Table 6-4. ICID connectivity (continued)**

IO Device	ICID Connectivity	
	IP Driven	SCFG Register
eDMA	-	One register to define 8 bits for ICID. Refer eDMA ICID register (DMA_ICID) in chapter "Supplemental Configuration Unit".
Debug path	-	One register to define 8 bits for ICID. Refer Debug ICID register (Debug_ICID) in chapter "Supplemental Configuration Unit".
eSDHC	-	One register to define 8 bits for ICID. Refer eSDHC ICID register (SDHC_ICID) in chapter "Supplemental Configuration Unit".

All the SMMU's support cache coherency for the page table walks and DVM transactions for page table cache maintenance operations. The PAGESIZE for all the SMMU's are 64 KB size.

The key guideline in the SMMU structure is that any transaction from any IP to a memory location (either to another IP or to the external memory) must go through the SMMU. This is true even if the device generates transactions using physical address, as the SMMU is also required for resource separation between VMs and must therefore inspect every transaction.

The following SMMU registers are implementation specific and their values are mentioned below:

Register	Secure access	Non-secure access
SMMU_IDR0	32'hFC01_7E40	32'h7C01_7E40
SMMU_IDR1	32'h4000_1F20	32'h4000_0020
SMMU_IDR2	32'h0000_5555	32'h0000_5555
SMMU_IDR7	32'h0000_0021	32'h0000_0021

# **Chapter 7**

## **CSU, OCRAM, and MSCM**

### **7.1 Central Security Unit**

The CSU manages the system security policy for accessing device peripherals. The CSU allows trusted code to set individual security access privileges for each of the peripherals, using one of eight security access privilege levels.

CSU features include:

- Configuration of peripheral access permissions for those peripherals unable to control their own access permissions
  - Note that some peripherals/bus slaves have their own access control capabilities (ability to reject transactions from unauthorized masters).
- Optional locking of individual CSU settings until the next POR
- Additional general purpose security related control bits

The CSU controls four parameters, allowing for eight security access privilege levels.

- Secure World vs Non-Secure World access
- Supervisor vs User access
- Read access
- Write access

Supervisor access is superset access to both User and Supervisor. Secure World access is superset access to both Non-Secure and Secure World.

The OCRAM is 64KB of On-Chip SRAM, restricted to Secure World access only by the CSU.

CSU related platform control registers are found in the Miscellaneous System Control Module. The MSCM provides error reporting related to attempted access control violations blocked by the CSU.

## 7.1.1 CSU Memory Map/Register Definition

The following registers, CSL and SA are collectively referred to as CSU Security Control Registers (SCRs) and can be written only by software executing in the TrustZone Secure Supervisor Mode.

**CSU memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
151_0000	Config Security Level (CSU_CSL)	32	R/W	0033_0033h	<a href="#">7.1.1.1/188</a>
151_0218	Secure Access register (CSU_SA0)	32	R/W	0000_0000h	<a href="#">7.1.1.2/194</a>
151_021C	Secure Access register (CSU_SA1)	32	R/W	0000_0000h	<a href="#">7.1.1.2/194</a>

### 7.1.1.1 Config Security Level (CSU\_CSL)

Each Config Security Level Register holds the config security level bits (access permission bits) for two targets. In the below table all peripherals are initially Secure User/Supervisor Read/Write Enabled, with no access by non-secure world masters. The CSL registers allow secure world to make the various peripherals' registers accessible to non-secure world software and bus masters.

#### NOTE

The PCI express, IFC, QuadSPI, eSDHC, SATA, USB, FMAN, QMAN, BMAN, and FlexTimer modules must be configured as non-secure peripherals and their default value needs to be configured as SL7-SL0 for all accesses enabled.

**Table 7-1. CSL Device Peripherals**

Offset	CSL Register[bits]	Reset value (in binary)	Peripheral
0xBASE_0000	CSL0[24:16]	0_0011_0011	PCI express controller 2 IO configuration space and memory space
	CSL0[8:0]	0_0011_0011	PCI express controller 1 IO configuration space and memory space
0xBASE_0004	CSL1[24:16]	0_0011_0011	Reserved
	CSL1[8:0]	0_0011_0011	Integrated Flash Controller (IFC) memory space
0xBASE_0008	CSL2[24:16]	0_0011_0011	OCRAM1
	CSL2[8:0]	0_0011_0011	Reserved
0xBASE_000C	CSL3[24:16]	0_0011_0011	PCI express controller 1 registers (including LUT)

*Table continues on the next page...*

**Table 7-1. CSL Device Peripherals (continued)**

Offset	CSL Register[bits]	Reset value (in binary)	Peripheral
	CSL3[8:0]	0_0011_0011	OCRAM2
0xBASE_0010	CSL4[24:16]	0_0011_0011	QuadSPI (QSPI) memory space
	CSL4[8:0]	0_0011_0011	PCI express controller 2 registers (including LUT)
0xBASE_0014	CSL5[24:16]	0_0011_0011	SATA controller registers
	CSL5[8:0]	0_0011_0011	USB 3.0 controllers registers USB controller 1 and PHY registers
0xBASE_0018	CSL6[24:16]	0_0011_0011	QMan and BMan Software portal
	CSL6[8:0]	0_0011_0011	Reserved
0xBASE_001C	CSL7[24:16]	0_0011_0011	Reserved
	CSL7[8:0]	0_0011_0011	Reserved
0xBASE_0020	CSL8[24:16]	0_0011_0011	PCI express controller 3 CCSR registers (including LUT)
	CSL8[8:0]	0_0011_0011	PCI express controller 3 (this includes memory, IO and Configuration space)
0xBASE_0024	CSL9[24:16]	0_0011_0011	Reserved
	CSL9[8:0]	0_0011_0011	Reserved
0xBASE_0028	CSL10[24:16]	0_0011_0011	USB3 controller and PHY registers (CCSR)
	CSL10[8:0]	0_0011_0011	USB controller 2 and PHY registers (CCSR)
0xBASE_002C	CSL11[24:16]	0_0011_0011	Reserved
	CSL11[8:0]	0_0011_0011	Reserved
0xBASE_0030	CSL12[24:16]	0_0011_0011	Reserved
	CSL12[8:0]	0_0011_0011	Reserved
0xBASE_0034	CSL13[24:16]	0_0011_0011	Reserved
	CSL13[8:0]	0_0011_0011	Reserved
0xBASE_0038	CSL14[24:16]	0_0011_0011	Reserved
	CSL14[8:0]	0_0011_0011	Reserved
0xBASE_003C	CSL15[24:16]	0_0011_0011	Reserved
	CSL15[8:0]	0_0011_0011	Reserved
0xBASE_0040	CSL16[24:16]	0_0011_0011	SerDes registers
	CSL16[8:0]	0_0011_0011	qDMA registers
0xBASE_0044	CSL17[24:16]	0_0011_0011	Low Power UART (LPUART) controller 2 registers
	CSL17[8:0]	0_0011_0011	Low Power UART (LPUART) controller 1 registers

*Table continues on the next page...*

**Table 7-1. CSL Device Peripherals (continued)**

Offset	CSL Register[bits]	Reset value (in binary)	Peripheral
0xBASE_0048	CSL18[24:16]	0_0011_0011	Low Power UART (LPUART) controller 4 registers
	CSL18[8:0]	0_0011_0011	Low Power UART (LPUART) controller 3 registers
0xBASE_004C	CSL19[24:16]	0_0011_0011	Low Power UART (LPUART) controller 6 registers
	CSL19[8:0]	0_0011_0011	Low Power UART (LPUART) controller 5 registers
0xBASE_0050	CSL20[24:16]	0_0011_0011	Reserved
	CSL20[8:0]	0_0011_0011	De-serial serial peripheral interface 1 (SPI1) registers
0xBASE_0054	CSL21[24:16]	0_0011_0011	QuadSPI (QSPI) controller registers
	CSL21[8:0]	0_0011_0011	Enhanced Secured Digital Host Controller (eSDHC) registers
0xBASE_0058	CSL22[24:16]	0_0011_0011	Reserved
	CSL22[8:0]	0_0011_0011	Integrated Flash Controller (IFC) registers
0xBASE_005C	CSL23[24:16]	0_0011_0011	I <sup>2</sup> C Controller 1 registers
	CSL23[8:0]	0_0011_0011	Reserved
0xBASE_0060	CSL24[24:16]	0_0011_0011	I <sup>2</sup> C Controller 3 registers
	CSL24[8:0]	0_0011_0011	I <sup>2</sup> C Controller 2 registers
0xBASE_0064	CSL25[24:16]	0_0011_0011	DUART 2 registers
	CSL25[8:0]	0_0011_0011	DUART 1 registers
0xBASE_0068	CSL26[24:16]	0_0011_0011	Watchdog 2 registers
	CSL26[8:0]	0_0011_0011	Watchdog 1 registers
0xBASE_006C	CSL27[24:16]	0_0011_0011	Enhanced DMA (eDMA) registers
	CSL27[8:0]	0_0011_0011	System Counter registers?
0xBASE_0070	CSL28[24:16]	0_0011_0011	Direct Memory Access Multiplexer 2
	CSL28[8:0]	0_0011_0011	Direct Memory Access Multiplexer 1
0xBASE_0074	CSL29[24:16]	0_0011_0011	DDR controller registers
	CSL29[8:0]	0_0011_0011	QUICC Engine registers
0xBASE_0078	CSL30[24:16]	0_0011_0011	DCFG, CCU, RCPM
	CSL30[8:0]	0_0011_0011	Secure Boot ROM controller registers
0xBASE_007C	CSL31[24:16]	0_0011_0011	Security Fuse Processor (SFP) registers
	CSL31[8:0]	0_0011_0011	Thermal Monitoring Unit (TMU) registers
0xBASE_0080	CSL32[24:16]	0_0011_0011	Security Monitor registers

Table continues on the next page...

**Table 7-1. CSL Device Peripherals (continued)**

Offset	CSL Register[bits]	Reset value (in binary)	Peripheral
	CSL32[8:0]	0_0011_0011	Supplemental configuration unit (SCFG)
0xBASE_0084	CSL33[24:16]	0_0011_0011	Frame Manager (FMan) registers
	CSL33[8:0]	0_0011_0011	SEC 5.5 registers
0xBASE_0088	CSL34[24:16]	0_0011_0011	Buffer Manager(BMan) registers
	CSL34[8:0]	0_0011_0011	Queue Manager(QMan) registers
0xBASE_008C	CSL35[24:16]	0_0011_0011	General Purpose IO (GPIO) controller 2 registers
	CSL35[8:0]	0_0011_0011	General Purpose IO (GPIO) controller 1 registers
0xBASE_0090	CSL36[24:16]	0_0011_0011	General Purpose IO (GPIO) controller 4 registers
	CSL36[8:0]	0_0011_0011	General Purpose IO (GPIO) controller 3 registers
0xBASE_0094	CSL37[24:16]	0_0011_0011	Platform Control registers Similar to COP SCFG and stores the status and address attributes of the blocked/failed transactions to the modules
	CSL37[8:0]	0_0011_0011	Central Security Unit (CSU) registers
0xBASE_0098	CSL38[24:16]	0_0011_0011	Reserved
	CSL38[8:0]	0_0011_0011	I <sup>2</sup> C Controller 4 registers
0xBASE_009C	CSL39[24:16]	0_0011_0011	Watchdog 4 registers
	CSL39[8:0]	0_0011_0011	Watchdog 3 registers
0xBASE_00A0	CSL40[24:16]	0_0011_0011	Reserved
	CSL40[8:0]	0_0011_0011	Watchdog 5 registers
0xBASE_00A4	CSL41[24:16]	0_0011_0011	Reserved
	CSL41[8:0]	0_0011_0011	Reserved
0xBASE_00A8	CSL42[24:16]	0_0011_0011	Reserved
	CSL42[8:0]	0_0011_0011	Reserved
0xBASE_00AC	CSL43[24:16]	0_0011_0011	Flex Timer Module 2 controller registers
	CSL43[8:0]	0_0011_0011	Flex Timer Module 1 controller registers
0xBASE_00B0	CSL44[24:16]	0_0011_0011	FlexTimer Module 4 controller registers
	CSL44[8:0]	0_0011_0011	Flex Timer Module 3 controller registers
0xBASE_00B4	CSL45[24:16]	0_0011_0011	FlexTimer Module 6 controller registers

*Table continues on the next page...*

**Table 7-1. CSL Device Peripherals (continued)**

Offset	CSL Register[bits]	Reset value (in binary)	Peripheral
	CSL45[8:0]	0_0011_0011	Flex Timer Module 5 controller registers
0xBASE_00B8	CSL46[24:16]	0_0011_0011	FlexTimer Module 8 controller registers
	CSL46[8:0]	0_0011_0011	FlexTimer Module 7 controller registers
CSL47-CSL59			Reserved
CSL60_REG[8:0]			DCSR

Address: 151\_0000h base + 0h offset = 151\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved							L2	SL15	SL14	SL13	SL12	SL11	SL10	SL9	SL8
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved							L1	SL7	SL6	SL5	SL4	SL3	SL2	SL1	SL0
Reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

**CSU\_CSL field descriptions**

Field	Description
31–25 -	This field is reserved.
24 L2	Lock bit corresponding to the slave. (It is written by secure software.) 0 Not locked. Bits 16-23 can be written by software. 1 Locked. Bits 16-23 cannot be written by software.
23 SL15	SL15 0 Non-secured supervisor write access disabled 1 Non-secured supervisor write access enabled
22 SL14	SL14 0 Non-secured user write access disabled 1 Non-secured user write access enabled
21 SL13	SL13 0 Secured supervisor write access disabled 1 Secured supervisor write access enabled
20 SL12	SL12 0 Secured user write access disabled 1 Secured user write access enabled
19 SL11	SL11

Table continues on the next page...

**CSU\_CSL field descriptions (continued)**

Field	Description
	0 Non-secured supervisor read access disabled 1 Non-secured supervisor read access enabled
18 SL10	SL10 0 Non-secured user read access disabled 1 Non-secured user read access enabled
17 SL9	SL9 0 Secured supervisor read access disabled 1 Secured supervisor read access enabled
16 SL8	SL8 0 Secured user read access disabled 1 Secured user read access enabled
15–9 -	This field is reserved.
8 L1	Lock bit corresponding to the slave. (It is written by secure software.) 0 Not locked. Bits 0-7 can be written by software. 1 Locked. Bits 0-7 cannot be written by software.
7 SL7	SL7 0 Non-secured supervisor write access disabled 1 Non-secured supervisor write access enabled
6 SL6	SL6 0 Non-secured user write access disabled 1 Non-secured user write access enabled
5 SL5	SL5 0 Secured supervisor write access disabled 1 Secured supervisor write access enabled
4 SL4	SL4 0 Secured user write access disabled 1 Secured user write access enabled
3 SL3	SL3 0 Non-secured supervisor read access disabled 1 Non-secured supervisor read access enabled
2 SL2	SL2 0 Non-secured user read access disabled 1 Non-secured user Read access enabled
1 SL1	SL1 0 Secured supervisor read access disabled 1 Secured supervisor read access enabled

*Table continues on the next page...*

**CSU\_CSL field descriptions (continued)**

Field	Description
0 SL0	SL0 0 Secured user read access disabled 1 Secured user read access enabled

**7.1.1.2 Secure Access register (CSU\_SAn)**

The following table provides the mapping of CSU secure access register bits.

**Table 7-2. Mapping of CSU secure access register bits**

Offset	Register Bits	Reset	Description
0xBASE_218	csu_sa0[5-4]	0x0	Software-override bit for SPNIDEN signal connectivity to core, CCI-400, SMMU, and debug components.
0xBASE_218	csu_sa0[7-6]	0x0	Software-override bit for NIDEN signal connectivity to core, CCI-400, SMMU, and debug components.
0xBASE_218	csu_sa0[9-8]	0x0	Software-override bit for SPIDEN signal connectivity to core, SMMU, and debug components
0xBASE_218	csu_sa0[11-10]	0x0	Software-Override bit for DBGEN signal connectivity to core, SMMU, and debug components
0xBASE_218	csu_sa0[13-12]	0x0	CP15SDISABLE for core 0
0xBASE_218	csu_sa0[15-14]	0x0	CP15SDISABLE for core 1
0xBASE_218	csu_sa0[17-16]	0x0	CFGDISABLE for GIC-400
0xBASE_218	csu_sa0[19-18]	0x0	Non-secure attribute control for debug components
0xBASE_218	csu_sa0[23-20]	0x0	-
0xBASE_218	csu_sa0[25-24]	0x0	CP15DISABLE for core2-
0xBASE_218	csu_sa0[27-26]	0x0	CP15DISABLE for core3-
0xBASE_218	csu_sa0[31-28]	0x0	-
0xBASE_21C	csu_sa1[3-2]	0x0	Trust Zone address space controller bypass mux. The TZASC module is disabled and logically bypassed.
0xBASE_21C	csu_sa1[5-4]	0x0	Secure boot lock for Trust-Zone address space controller registers

Address: 151\_0000h base + 218h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	L15_n	SA15_n	L14_n	SA14_n	L13_n	SA13_n	L12_n	SA12_n	L11_n	SA11_n	L10_n	SA10_n	L9_n	SA9_n	L8_n	SA8_n
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L7_n	SA7_n	L6_n	SA6_n	L5_n	SA5_n	L4_n	SA4_n	L3_n	SA3_n	L2_n	SA2_n	L1_n	SA1_n	L0_n	SA0_n
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### CSU\_SAn field descriptions

Field	Description
31 L15_n	Lock bit set by secure software. 0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
30 SA15_n	Secured access indicator 0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
29 L14_n	Lock bit set by secure software. 0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
28 SA14_n	Secured access indicator 0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
27 L13_n	Lock bit set by secure software. 0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
26 SA13_n	Secured access indicator 0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
25 L12_n	Lock bit set by secure software. 0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
24 SA12_n	Secured access indicator 0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.

Table continues on the next page...

**CSU\_SAn field descriptions (continued)**

Field	Description
23 L11_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
22 SA11_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
21 L10_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
20 SA10_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
19 L9_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
18 SA9_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
17 L8_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
16 SA8_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
15 L7_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
14 SA7_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
13 L6_n	Lock bit set by secure software.  0 Stands for no lock condition and (2*n)th bit can be written by software. 1 Stands for locking of (2*n) th bit and once set software cannot write on the SA field.
12 SA6_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.

*Table continues on the next page...*

**CSU\_SAn field descriptions (continued)**

Field	Description
11 L5_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
10 SA5_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
9 L4_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
8 SA4_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
7 L3_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
6 SA3_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
5 L2_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
4 SA2_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
3 L1_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
2 SA1_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.
1 L0_n	Lock bit set by secure software.  0 Stands for no lock condition and $(2^n)$ th bit can be written by software. 1 Stands for locking of $(2^n)$ th bit and once set software cannot write on the SA field.
0 SA0_n	Secured access indicator  0 Stands for Secured Access indication for nth Type 1 Master corresponding to this 32-bit register. 1 Stands for Non-Secured Access for that master.

## 7.1.2 Initialization Policy

For peripherals whose access controls are configured in the CSU

1. Set (and optionally lock) the CSU\_CSLn register fields to set each peripheral's access permissions as a target
2. Set (and optionally lock) the CSU\_SA register fields

### NOTE

Once set, CSU lock bits can only be cleared by a device reset.

## 7.2 On-Chip RAM memory controller (OCRAM)

The on-chip RAM is implemented as a slave device on the 64-bit system AXI bus. There are two OCRAMs in the chip and access can be controlled through CSU peripheral register programming.

Each OCRAM occupies a 64 KB of address region and the start address of each OCRAM (64 KB each) in the memory map is given below:

- OCRAM1: 0x1000\_0000
- OCRAM2: 0x1001\_0000.

### NOTE

The OCRAM1 and OCRAM2 memory content is not initialized to zero by the chip. The software must initialize both OCRAM1 and OCRAM2 in full width (64-bit) access and then perform a write to bit 5 and 6 at address 0x2014\_0534 and 0x2014\_0544, respectively with the value 0x0 to clear the single and multi-bit ECC errors for OCRAM1 and OCRAM2. Also, the ECC error interrupt needs to be cleared in GIC (MBEE). Once completed, the host processor enables interrupt related to ECC errors from OCRAM1 and OCRAM2

## 7.3 Miscellaneous System Control Module (MSCM)

The MSCM (platform control) stores the status and address attributes of a transaction to a peripheral which was blocked by the CSU. The MSCM also enables CSU interrupts.

### 7.3.1 MSCM Access Control and TrustZone Security (ACTZS)Memory Map/Register Definition

The ACTZS configuration portion of the MSCM programming model map is shown in the table below. It is partitioned into two sections:

Offset addresses 0xC00 - 0xC18 define interrupt configurability of CSU related access violation reporting.

Offset addresses 0xD00 - 0xDDC contain captured access address and attribute information for CSL-detected violations.

Attempted writes to read-only registers are simply ignored (RO/WI). This section contains the target access fault information like CSL $n$  attribute check logic plus an array of 128-bit register structures containing captured CSL $n$  fault information.

All the register accesses are privilege/supervisor

**MSCM\_ACTZS memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
152_0C10	ACTZS CSL Interrupt Enable Register (MSCM_ACTZS_CSLIER)	32	R/W	0000_0000h	<a href="#">7.3.1.1/199</a>
152_0C14	ACTZS CSL Interrupt Register (MSCM_ACTZS_CSLIR)	32	R/W	0000_0000h	<a href="#">7.3.1.2/202</a>
152_0C18	ACTZS CSL Interrupt Overrun Register (MSCM_ACTZS_CSOPR)	32	R/W	0000_0000h	<a href="#">7.3.1.3/205</a>

#### 7.3.1.1 ACTZS CSL Interrupt Enable Register (MSCM\_ACTZS\_CSLIER)

The CSL $n$  interrupt enable register is a 32-bit register containing a bit map of interrupt enables for CSL $n$  logic reporting access check violations.

As the individual CSL $n$  levels are evaluated for each bus transfer, access violations are error terminated and the resulting error status flags collected and posted in the MSCM\_CSLIR.

The CSLIER also contains a lock bit (RO) that may be set to disable writes to the register, preserving the enabled/disabled state of the CSL $n$  interrupts.

## Miscellaneous System Control Module (MSCM)

Address: 152\_0000h base + C10h offset = 152\_0C10h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
Lock																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
CIE15																
Reserved																
CIE13																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MSCM\_ACTZS\_CSLIER field descriptions

Field	Description
0 Lock	Read-Only. This register bit provides a mechanism to "lock" the configuration state defined by MSCM_CSLIER. Once asserted, attempted writes to the MSCM_CSLIER register are ignored until the next system reset clears the flag.  0 writes to the MSCM_CSLIER are allowed  1 writes to the MSCM_CSLIER are ignored
1–9 -	This field is reserved. This field is reserved
10 CIE21	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt.  if CIE21= 0, the CSLn access check interrupt is disabled.  if CIE21= 1, the CSLn access check interrupt is enabled.  The individual CIE21 are mapped to QMAN software portal.
11 CIE20	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt.  if CIE20= 0, the CSLn access check interrupt is disabled.  if CIE20= 1, the CSLn access check interrupt is enabled.  The individual CIE20 are mapped to USB 3 Controller 3 Register space(CCSR).
12 CIE19	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt.  if CIE19= 0, the CSLn access check interrupt is disabled.  if CIE19= 1, the CSLn access check interrupt is enabled.  The individual CIE19 are mapped to USB 3Controller 2 Register space(CCSR).

Table continues on the next page...

**MSCM\_ACTZS\_CSLIER field descriptions (continued)**

Field	Description
13 CIE18	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE18 = 0, the CSLn access check interrupt is disabled. if CIE18 = 1, the CSLn access check interrupt is enabled. The individual CIE18 are mapped to OCRAM2 memory space.
14 CIE17	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE17 = 0, the CSLn access check interrupt is disabled. if CIE17 = 1, the CSLn access check interrupt is enabled. The individual CIE17 are mapped to OCRAM1 memory space.
15 CIE16	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE16= 0, the CSLn access check interrupt is disabled. if CIE16= 1, the CSLn access check interrupt is enabled. The individual CIE16 are mapped to SATA Register space(CCSR).
16 CIE15	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE15= 0, the CSLn access check interrupt is disabled. if CIE15= 1, the CSLn access check interrupt is enabled. The individual CIE15 are mapped to PCI Express memory map space.
17 -	This field is reserved. Reserved
18 CIE13	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE13= 0, the CSLn access check interrupt is disabled. if CIE13= 1, the CSLn access check interrupt is enabled. The individual CIE13 are mapped to GIC Register space (CCSR).
19 CIE12	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE12= 0, the CSLn access check interrupt is disabled. if CIE12= 1, the CSLn access check interrupt is enabled. The individual CIE12 are mapped to USB3 Controller Register space(CCSR).
20 -	This field is reserved. Reserved
21 CIE10	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE10= 0, the CSLn access check interrupt is disabled. if CIE10= 1, the CSLn access check interrupt is enabled. The individual CIE10 are mapped to SATA Register space(CCSR).
22 CIE9	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE9= 0, the CSLn access check interrupt is disabled. if CIE9= 1, the CSLn access check interrupt is enabled. The individual CIE9 are mapped to IFC Memory space.
23 CIE8	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE8 = 0, the CSLn access check interrupt is disabled.

*Table continues on the next page...*

**MSCM\_ACTZS\_CSLIER field descriptions (continued)**

Field	Description
	if CIE8 = 1, the CSLn access check interrupt is enabled. The individual CIE8 are mapped to Register Configuration Space of all the IP modules in CCSR Space except PCI Express controllers, SATA controller, USB 3 Controller Register configuration space.
24–26 -	This field is reserved. Reserved
27 CIE4	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE4 = 0, the CSLn access check interrupt is disabled. if CIE4 = 1, the CSLn access check interrupt is enabled. The individual CIE4 are mapped to QuadSPI memory space.
28 CIE3	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE3 = 0, the CSLn access check interrupt is disabled. if CIE3 = 1, the CSLn access check interrupt is enabled. The individual CIE3 are mapped to PCI Express controller 2 Register space(CCSR).
29 CIE2	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE2 = 0, the CSLn access check interrupt is disabled. if CIE2 = 1, the CSLn access check interrupt is enabled. The individual CIE2 are mapped to PCI Express controller 1 Register space(CCSR).
30 CIE1	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE1 = 0, the CSLn access check interrupt is disabled. if CIE1 = 1, the CSLn access check interrupt is enabled. The individual CIE1 are mapped to PCI Express controller 2 IO Config Space and memory space.
31 CIE0	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if CIE0 = 0, the CSLn access check interrupt is disabled. if CIE0 = 1, the CSLn access check interrupt is enabled. The individual CIE0 are mapped to PCI Express controller 1 IO Config Space and memory space.

**7.3.1.2 ACTZS CSL Interrupt Register (MSCM\_ACTZS\_CSLIR)**

The CSLn interrupt register is a 32-bit register containing a bit map of interrupts for CSLn logic reporting access check violations.

As the individual CSLn levels are evaluated for each bus transfer, access violations are error terminated and the resulting error status flags collected and posted in the MSCM\_CSLIR.

The MSCM\_CSLIR records all CSLn access violations regardless of the state of the interrupt enable (MSCM\_CSLIER). Accordingly, this register can be used by both interrupt service routines and/or bus error exception handlers to quickly determine the source of the CSLn access check violation. Each individual interrupt flag in this register

is cleared by writing a "1" to it; this would typically be done after the captured fail address and attribute information is retrieved from the appropriate MSCM\_CSF\*R register. Additionally, the clearing of an interrupt flag in this register also clears the corresponding bit in the MSCM\_CSLOVR register and rearms the logic for capturing the failed access information.

Address: 152\_0000h base + C14h offset = 152\_0C14h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MSCM\_ACTZS\_CSLIR field descriptions

Field	Description
0–17 -	This field is reserved. This field is reserved
18 INT13	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT13= 0, the CSLn access check interrupt is disabled. if INT13 = 1, the CSLn access check interrupt is enabled. The individual CIE13 are mapped to GIC Register space(CCSR).
19 INT12	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT12= 0, the CSLn access check interrupt is disabled. if INT12 = 1, the CSLn access check interrupt is enabled. The individual CIE12 are mapped to USB 3 Controller Register space(CCSR).
20 -	This field is reserved. Reserved
21 INT10	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT10 = 0, the CSLn access check interrupt is disabled.

Table continues on the next page...

**MSCM\_ACTZS\_CSLIR field descriptions (continued)**

Field	Description
	if INT10 = 1, the CSLn access check interrupt is enabled. The individual CIE10 are mapped to SATA register space(CCSR).
22 INT9	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT9 = 0, the CSLn access check interrupt is disabled. if INT9 = 1, the CSLn access check interrupt is enabled. The individual CIE9 are mapped to IFC memory space.
23 INT8	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT8 = 0, the CSLn access check interrupt is disabled. if INT8 = 1, the CSLn access check interrupt is enabled. The individual CIE8 are mapped to Register Configuration Space of all the IP modules in CCSR Space except PCI Express controllers, , SATA controller, USB 3 Controller Register configuration space.
24–26 -	This field is reserved. Reserved
27 INT4	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT4 = 0, the CSLn access check interrupt is disabled. if INT4 = 1, the CSLn access check interrupt is enabled. The individual CIE4 are mapped to QuadSPI memory space.
28 INT3	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT3 = 0, the CSLn access check interrupt is disabled. if INT3 = 1, the CSLn access check interrupt is enabled. The individual CIE3 are mapped to PCI Express controller 2 Register space (CCSR).
29 INT2	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT2 = 0, the CSLn access check interrupt is disabled. if INT2 = 1, the CSLn access check interrupt is enabled. The individual CIE2 are mapped to PCI Express controller 1 Register space (CCSR).
30 INT1	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT1 = 0, the CSLn access check interrupt is disabled. if INT1 = 1, the CSLn access check interrupt is enabled. The individual CIE1 are mapped to PCI Express controller 2 IO config space and memory space.
31 INT0	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if INT0 = 0, the CSLn access check interrupt is disabled. if INT0 = 1, the CSLn access check interrupt is enabled. The individual CIE0 are mapped to PCI Express controller 1 IO config space and memory space.

### 7.3.1.3 ACTZS CSL Interrupt Overrun Register (MSCM\_ACTZS\_CSOVR)

The CSL $n$  interrupt overrun register is a 32-bit read-only register containing a bit map of overrun interrupt conditions for the CSL $n$  logic reporting access check violations.

The overrun condition is simply defined as the detection of another CSL $n$  access check violation before the previous one has been full processed and cleared. Stated differently, if a CSL $n$  access check violation is detected and the corresponding MSCM\_CSLIR[i] bit still asserted, the MSCM\_CSOVR[i] bit is set.

Additionally, for the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSL $n$  Fail Status Registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and set the appropriate overrun indicator in the MSCM\_CSOVR.

Each individual interrupt flags is cleared by writing a "1" to it and this is typically be done after the captured fail address and attribute information is retrieved from the appropriate MSCM\_CSF\*R register. The clearing of an interrupt flag in the MSCM\_CSLIR also clears the corresponding bit in the MSCM\_CSOVR register and rearms the logic for capturing the fail access information.

Address: 152\_0000h base + C18h offset = 152\_0C18h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R			Reserved	OVR13	Reserved	OVR10	OVR9	OVR8		Reserved	OVR4	OVR3	OVR2	OVR1	OVR0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MSCM\_ACTZS\_CSOVR field descriptions**

<b>Field</b>	<b>Description</b>
0–17 -	This field is reserved. This field is reserved.
18 OVR13	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR13= 0, the CSLn access check interrupt is disabled. if OVR13 = 1, the CSLn access check interrupt is enabled. The individual CIE13 are mapped to GIC Register space(CCSR).
19 OVR12	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR12= 0, the CSLn access check interrupt is disabled. if OVR12 = 1, the CSLn access check interrupt is enabled. The individual CIE12 are mapped to USB 3 Controller Register space(CCSR).
20 -	This field is reserved. Reserved
21 OVR10	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR10 = 0, the CSLn access check interrupt is disabled. if OVR10 = 1, the CSLn access check interrupt is enabled. The individual CIE10 are mapped to SATA Register space(CCSR).
22 OVR9	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR9 = 0, the CSLn access check interrupt is disabled. if OVR9 = 1, the CSLn access check interrupt is enabled. The individual CIE9 are mapped to IFC Memory space.
23 OVR8	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR8 = 0, the CSLn access check interrupt is disabled. if OVR8 = 1, the CSLn access check interrupt is enabled. The individual CIE8 are mapped to Register Configuration Space of all the IP modules in CCSR Space except PCI Express controllers, , SATA controller, USB 3 Controller Register config space.
24–26 -	This field is reserved. Reserved
27 OVR4	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR4 = 0, the CSLn access check interrupt is disabled. if OVR4 = 1, the CSLn access check interrupt is enabled. The individual CIE4 are mapped to QuadSPI memory space.
28 OVR3	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR3 = 0, the CSLn access check interrupt is disabled. if OVR3 = 1, the CSLn access check interrupt is enabled. The individual CIE3 are mapped to PCI Express controller 2 Register space(CCSR).
29 OVR2	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR2 = 0, the CSLn access check interrupt is disabled. if OVR2 = 1, the CSLn access check interrupt is enabled. The individual CIE2 are mapped to PCI Express controller 1 Register space(CCSR).

*Table continues on the next page...*

**MSCM\_ACTZS\_CSOVR field descriptions (continued)**

Field	Description
30 OVR1	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR1 = 0, the CSLn access check interrupt is disabled. if OVR1 = 1, the CSLn access check interrupt is enabled. The individual CIE1 are mapped to PCI Express controller 2 IO config space and memory space.
31 OVR0	CSLn Interrupt Enable. This field enables/disables each CSLn access violation alert interrupt. if OVR0 = 0, the CSLn access check interrupt is disabled. if OVR0 = 1, the CSLn access check interrupt is enabled. The individual CIE0 are mapped to PCI Express controller 1 IO config space and memory space.

### 7.3.2 ACTZS CSLn Fail Status Capture Registers (Memory Map/ Register Definition)

This section of the MSCM\_ACTZS programming model contains an array of four word (128-bit) data values containing address and attribute information corresponding to CSLn access check violations. The format of this data structure is identical to the fail status information captured by the TZASC when they detect a security violation.

When a CSLn access check violation is detected, the bus transaction is error terminated, the appropriate bit in the MSCM\_CSLIR set and the fail address and attribute information captured in the corresponding data structure. The contents of the captured fail data is unaffected until the interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

The LS1 implementation supports n = [0-14] and contains an array of ten 128-bit data structures and four reserved structures as defined in table below.

**Table 7-3. MSCM CSLn Fail Status Capture Registers**

Base Offset Address	Source
0xD00	PCI Express controller 1 IO Config Space and memory space
0xD10	PCI Express controller 2 IO Config Space and memory space
0xD20	PCI Express controller 1 Register space(CCSR)
0xD30	PCI Express controller 2 Register space(CCSR)
0xD40	QuadSPI memory space
0xD50-0xD70	Reserved

*Table continues on the next page...*

**Table 7-3. MSCM CSLn Fail Status Capture Registers  
(continued)**

Base Offset Address	Source
0xD80	Register configuration space of all the IP modules in CCSR Space except PCI Express controllers, SATA controller, USB 3 controller register config space
0xD90	IFC Memory space
0xDA0	SATA Register space(CCSR)
0xDB0	Reserved
0xDC0	USB 3 controller1 Register space(CCSR)
0xDD0-0xDE0	Reserved
0xDF0	PCI Express controller 3 IO Config Space and memory space
0xE00	PCI Express controller 3 Register space(CCSR)
0xE10	OCRAM1 Memory Space
0xE20	OCRAM2 Memory Space
0xE30	USB Controller 2 CCSR register(CCSR)
0xE40	USB Controller 3 register space(CCSR)
0xE50	QMan/BMan Software portal memory space

All the register accesses are privilege/supervisor.

### MSCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
152_0D00	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR0)	32	R/W	0000_0000h	<a href="#">7.3.2.1/211</a>
152_0D08	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR0)	32	R/W	0000_0000h	<a href="#">7.3.2.2/212</a>
152_0D0C	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIG0)	32	R/W	0000_0000h	<a href="#">7.3.2.3/213</a>
152_0D10	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR1)	32	R/W	0000_0000h	<a href="#">7.3.2.1/211</a>
152_0D18	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR1)	32	R/W	0000_0000h	<a href="#">7.3.2.2/212</a>
152_0D1C	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIG1)	32	R/W	0000_0000h	<a href="#">7.3.2.3/213</a>
152_0D20	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR2)	32	R/W	0000_0000h	<a href="#">7.3.2.1/211</a>
152_0D28	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR2)	32	R/W	0000_0000h	<a href="#">7.3.2.2/212</a>

*Table continues on the next page...*

**MSCM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
152_0D2C	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR2)	32	R/W	0000_0000h	<a href="#">7.3.2.3/213</a>
152_0D30	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR3)	32	R/W	0000_0000h	<a href="#">7.3.2.1/211</a>
152_0D38	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR3)	32	R/W	0000_0000h	<a href="#">7.3.2.2/212</a>
152_0D3C	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR3)	32	R/W	0000_0000h	<a href="#">7.3.2.3/213</a>
152_0D40	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR4)	32	R/W	0000_0000h	<a href="#">7.3.2.1/211</a>
152_0D48	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR4)	32	R/W	0000_0000h	<a href="#">7.3.2.2/212</a>
152_0D4C	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR4)	32	R/W	0000_0000h	<a href="#">7.3.2.3/213</a>
152_0D80	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR8)	32	R/W	0000_0000h	<a href="#">7.3.2.4/214</a>
152_0D88	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR8)	32	R/W	0000_0000h	<a href="#">7.3.2.5/215</a>
152_0D8C	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR8)	32	R/W	0000_0000h	<a href="#">7.3.2.6/216</a>
152_0D90	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR9)	32	R/W	0000_0000h	<a href="#">7.3.2.4/214</a>
152_0D98	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR9)	32	R/W	0000_0000h	<a href="#">7.3.2.5/215</a>
152_0D9C	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR9)	32	R/W	0000_0000h	<a href="#">7.3.2.6/216</a>
152_0DA0	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR10)	32	R/W	0000_0000h	<a href="#">7.3.2.4/214</a>
152_0DA8	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR10)	32	R/W	0000_0000h	<a href="#">7.3.2.5/215</a>
152_0DAC	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR10)	32	R/W	0000_0000h	<a href="#">7.3.2.6/216</a>
152_0DC0	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR12)	32	R/W	0000_0000h	<a href="#">7.3.2.7/217</a>
152_0DC8	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR12)	32	R/W	0000_0000h	<a href="#">7.3.2.8/218</a>
152_0DCC	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR12)	32	R/W	0000_0000h	<a href="#">7.3.2.9/219</a>
152_0DF0	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CS FAR14)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0DF8	ACTZS CSLn Fail Status Master ID Register (MSCM_CS FIR14)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0DFC	ACTZS CSLn Fail Status Control Register (MSCM_CS FCR14)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>

Table continues on the next page...

**MSCM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
152_0E00	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR15)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E08	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR15)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E0C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR15)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>
152_0E10	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR16)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E18	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR16)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E1C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR16)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>
152_0E20	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR17)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E28	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR17)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E2C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR17)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>
152_0E30	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR18)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E38	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR18)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E3C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR18)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>
152_0E40	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR19)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E48	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR19)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E4C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR19)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>
152_0E50	ACTZS CSLn Fail Status Address (Low) Register (MSCM_CSFAR20)	32	R/W	0000_0000h	<a href="#">7.3.2.10/220</a>
152_0E58	ACTZS CSLn Fail Status Master ID Register (MSCM_CSFIR20)	32	R/W	0000_0000h	<a href="#">7.3.2.11/221</a>
152_0E5C	ACTZS CSLn Fail Status Control Register (MSCM_CSFCR20)	32	R/W	0000_0000h	<a href="#">7.3.2.12/221</a>

### 7.3.2.1 ACTZS CSLn Fail Status Address (Low) Register (MSCM\_CSFARn)

The CSFARn is a 32-bit read-only register for capturing the low-order 32 bits of the 40-bit physical address of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFARn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

#### NOTE

The next sequential word at offset address  $0xD04 + 16*n$  is reserved for systems where the address space is larger than 4 Gbytes.

Address: 152\_0000h base + D00h offset + (16d × i), where i=0d to 4d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0

#### MSCM\_CSFARn field descriptions

Field	Description
0–31 FAD	CSLn Fail Address. This read-only field specifies the system address from the last captured CSLn access check violation.

### 7.3.2.2 ACTZS CSLn Fail Status Control Register (MSCM\_CSFCRn)

The CSFCRn is a 32-bit read-only register for capturing specific attribute bits of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn Interrupt Register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn Fail Status Registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFCRn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + D08h offset + (16d × i), where i=0d to 4d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved							FWT	Reserved		FNS	FPR	Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved							Reserved								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MSCM\_CSFCRn field descriptions**

Field	Description
0–6 -	This field is reserved. Reserved
7 FWT	CSLn Fail Write. This read-only field specifies the read/write attribute from the last captured CSLn access check violation. if FWT = 0, then the last captured CSLn access check violation was a read. if FWT = 1, then the last captured CSLn access check violation was a write.
8–9 -	This field is reserved. Reserved
10 FNS	CSLn Fail Nonsecure. This read-only field specifies the secure/nonsecure attribute from the last captured CSLn access check violation. if FNS = 0, the last captured CSLn access check violation was a secure access. if FNS = 1, the last captured CSLn access check violation was a nonsecure access.

*Table continues on the next page...*

## **MSCM\_CSFCR*n* field descriptions (continued)**

Field	Description
11 FPR	<p>CSLn Fail Privileged. This read-only field specifies the user/privileged attribute from the last captured CSLn access check violation.</p> <p>if FPR = 0, the last captured CSLn access check violation was a user mode access.</p> <p>if FPR = 1, the last captured CSLn access check violation was a privileged access.</p>
12-31 -	<p>This field is reserved.</p> <p>Reserved</p>

### 7.3.2.3 ACTZS CSLn Fail Status Master ID Register (MSCM\_CSFI $n$ )

The CSFIRn is a 32-bit read-only register for capturing the 4 LSB bits of the AXI ID attribute of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFIRn is unaffected until the corresponding MSCM CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + D0Ch offset + (16d × i), where i=0d to 4d

## MSCM CSFIRn field descriptions

Field	Description
0-26 -	This field is reserved. Reserved
27-31 FMID	CSLn Fail Master ID. This read-only field specifies the master ID from the last captured CSLn access check violation.

### 7.3.2.4 ACTZS CSLn Fail Status Address (Low) Register (MSCM\_CSFARn)

The CSFARn is a 32-bit read-only register for capturing the low-order 32 bits of the 40-bit physical address of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFARn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

#### NOTE

The next sequential word at offset address  $0xD04 + 16*n$  is reserved for systems where the address space is larger than 4 Gbytes.

Address: 152\_0000h base + D80h offset + (16d × i), where i=0d to 2d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0

#### MSCM\_CSFARn field descriptions

Field	Description
0–31 FAD	CSLn Fail Address. This read-only field specifies the system address from the last captured CSLn access check violation.

### 7.3.2.5 ACTZS CSLn Fail Status Control Register (MSCM\_CSFCRn)

The CSFCRn is a 32-bit read-only register for capturing specific attribute bits of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn Interrupt Register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn Fail Status Registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFCRn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + D88h offset + (16d × i), where i=0d to 2d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved							FWT	Reserved	FNS	FPR	Reserved				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved							Reserved								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**MSCM\_CSFCRn field descriptions**

Field	Description
0–6 -	This field is reserved. Reserved
7 FWT	CSLn Fail Write. This read-only field specifies the read/write attribute from the last captured CSLn access check violation. if FWT = 0, then the last captured CSLn access check violation was a read. if FWT = 1, then the last captured CSLn access check violation was a write.
8–9 -	This field is reserved. Reserved
10 FNS	CSLn Fail Nonsecure. This read-only field specifies the secure/nonsecure attribute from the last captured CSLn access check violation. if FNS = 0, the last captured CSLn access check violation was a secure access. if FNS = 1, the last captured CSLn access check violation was a nonsecure access.

*Table continues on the next page...*

## **MSCM CSFCR*n* field descriptions (continued)**

Field	Description
11 FPR	<p>CSLn Fail Privileged. This read-only field specifies the user/privileged attribute from the last captured CSLn access check violation.</p> <p>if FPR = 0, the last captured CSLn access check violation was a user mode access.</p> <p>if FPR = 1, the last captured CSLn access check violation was a privileged access.</p>
12-31 -	<p>This field is reserved.</p> <p>Reserved</p>

### 7.3.2.6 ACTZS CSLn Fail Status Master ID Register (MSCM CSFIRn)

The CSFIRn is a 32-bit read-only register for capturing the 4 LSB bits of the AXI ID attribute of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFIRn is unaffected until the corresponding MSCM CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + D8Ch offset + (16d × i), where i=0d to 2d

MSCM CSFIBn field descriptions

Field	Description
0-26 -	This field is reserved. Reserved
27-31 FMID	CSLn Fail Master ID. This read-only field specifies the master ID from the last captured CSLn access check violation.

### 7.3.2.7 ACTZS CSLn Fail Status Address (Low) Register (MSCM\_CSFARn)

The CSFARn is a 32-bit read-only register for capturing the low-order 32 bits of the 40-bit physical address of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFARn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

#### NOTE

The next sequential word at offset address  $0xD04 + 16*n$  is reserved for systems where the address space is larger than 4 Gbytes.

Address: 152\_0000h base + DC0h offset + (16d × i), where i=0d to 0d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

#### MSCM\_CSFARn field descriptions

Field	Description
0–31 FAD	CSLn Fail Address. This read-only field specifies the system address from the last captured CSLn access check violation.

### 7.3.2.8 ACTZS CSLn Fail Status Control Register (MSCM\_CSFCR $n$ )

The CSFCR $n$  is a 32-bit read-only register for capturing specific attribute bits of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFAR $n$ , CSFCR $n$  and CSFIR $n$  registers, and the appropriate flag in the CSLn Interrupt Register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn Fail Status Registers (MSCM\_CSFAR $n$ , MSCM\_CSFCR $n$ , MSCM\_CSFIR $n$ ) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFCR $n$  is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + DC8h offset + (16d × i), where i=0d to 0d

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R W	Reserved							FWT	Reserved		FNS	FPR	Reserved				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R W	Reserved								Reserved								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### MSCM\_CSFCR $n$ field descriptions

Field	Description
0–6 -	This field is reserved. Reserved
7 FWT	CSLn Fail Write. This read-only field specifies the read/write attribute from the last captured CSLn access check violation.  if FWT = 0, then the last captured CSLn access check violation was a read.  if FWT = 1, then the last captured CSLn access check violation was a write.
8–9 -	This field is reserved. Reserved
10 FNS	CSLn Fail Nonsecure. This read-only field specifies the secure/nonsecure attribute from the last captured CSLn access check violation.  if FNS = 0, the last captured CSLn access check violation was a secure access.  if FNS = 1, the last captured CSLn access check violation was a nonsecure access.

Table continues on the next page...

## **MSCM\_CSFCR*n* field descriptions (continued)**

Field	Description
11 FPR	<p>CSLn Fail Privileged. This read-only field specifies the user/privileged attribute from the last captured CSLn access check violation.</p> <p>if FPR = 0, the last captured CSLn access check violation was a user mode access.</p> <p>if FPR = 1, the last captured CSLn access check violation was a privileged access.</p>
12-31 -	<p>This field is reserved.</p> <p>Reserved</p>

### 7.3.2.9 ACTZS CSLn Fail Status Master ID Register (MSCM\_CSFI $n$ )

The CSFIRn is a 32-bit read-only register for capturing the 4 LSB bits of the AXI ID attribute of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFIRn is unaffected until the corresponding MSCM CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + DCCh offset + (16d × i), where i=0d to Od

## MSCM CSFIRn field descriptions

Field	Description
0–26 - Reserved	This field is reserved.
27–31 FMID	CSLn Fail Master ID. This read-only field specifies the master ID from the last captured CSLn access check violation.

### 7.3.2.10 ACTZS CSLn Fail Status Address (Low) Register (MSCM\_CSFARn)

The CSFARn is a 32-bit read-only register for capturing the low-order 32 bits of the 40-bit physical address last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFARn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

#### NOTE

The next sequential word at offset address  $0xD04 + 16*n$  is reserved for systems where the address space is larger than 4 Gbytes.

Address: 152\_0000h base + DF0h offset + (16d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0

#### MSCM\_CSFARn field descriptions

Field	Description
0–31 FAD	CSLn Fail Address. This read-only field specifies the system address from the last captured CSLn access check violation.

### 7.3.2.11 ACTZS CSLn Fail Status Master ID Register (MSCM\_CSFIRn)

The CSFIRn is a 32-bit read-only register for capturing the 4 LSB bits of the AXI ID attribute of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn interrupt register (CSLIR) is asserted.

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn fail status registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the MSCM\_CSOVR.

The contents of the MSCM\_CSFIRn is unaffected until the corresponding MSCM\_CSLIR interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + DF8h offset + (16d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**MSCM\_CSFIRn field descriptions**

Field	Description
0–26 -	This field is reserved. Reserved
27–31 FMID	CSLn Fail Master ID. This read-only field specifies the master ID from the last captured CSLn access check violation.

### 7.3.2.12 ACTZS CSLn Fail Status Control Register (MSCM\_CSFCRn)

The CSFCRn is a 32-bit read-only register for capturing specific attribute bits of the last captured access check violation detected by the CSLn logic. When the CSLn logic detects an access violation, the address and attributes associated with the memory reference are loaded into the CSFARn, CSFCRn and CSFIRn registers, and the appropriate flag in the CSLn Interrupt Register (CSLIR) is asserted.

## Miscellaneous System Control Module (MSCM)

For the slave ACTZS ports using the AXI bus protocol, it is possible to detect simultaneous access violations on the read and write channels. For this condition, the CSLn Fail Status Registers (MSCM\_CSFARn, MSCM\_CSFCRn, MSCM\_CSFIRn) capture the information associated with the write access and additionally set the appropriate overrun indicator in the **MSCM\_CSOVR**.

The contents of the **MSCM\_CSFCRn** is unaffected until the corresponding **MSCM\_CSLIR** interrupt flag is cleared by writing a 1 to it, at which time, the capturing of fail information is rearmed.

Attempted privileged mode writes are ignored.

Address: 152\_0000h base + DFCh offset + (16d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved							FWT	Reserved	FNS	FPR	Reserved				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MSCM\_CSFCRn field descriptions

Field	Description
0–6 -	This field is reserved. Reserved
7 FWT	CSLn Fail Write. This read-only field specifies the read/write attribute from the last captured CSLn access check violation.  if FWT = 0, then the last captured CSLn access check violation was a read. if FWT = 1, then the last captured CSLn access check violation was a write.
8–9 -	This field is reserved. Reserved
10 FNS	CSLn Fail Nonsecure. This read-only field specifies the secure/nonsecure attribute from the last captured CSLn access check violation.  if FNS = 0, the last captured CSLn access check violation was a secure access. if FNS = 1, the last captured CSLn access check violation was a nonsecure access.
11 FPR	CSLn Fail Privileged. This read-only field specifies the user/privileged attribute from the last captured CSLn access check violation.  if FPR = 0, the last captured CSLn access check violation was a user mode access. if FPR = 1, the last captured CSLn access check violation was a privileged access.
12–31 -	This field is reserved. Reserved

# Chapter 8

## System Counter

### 8.1 System counter

The system counter implements the Arm Generic Timer requirements mentioned in the chapter "System Level Implementation of the Generic Timer" of Armv8, for Armv8-A architecture profile.

#### 8.1.1 Secure system counter memory map/register definition

This table shows the register list of CCSR access control unit.

**Secure\_system\_counter memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2B0_0000	Control register (Secure_system_counter_CNTCR)	32	R/W	0000_0000h	<a href="#">8.1.1.1/224</a>
2B0_0008	LSB of counter count value register (Secure_system_counter_CNTCV1)	32	R/W	0000_0000h	<a href="#">8.1.1.2/224</a>
2B0_000C	MSB of counter count value register (Secure_system_counter_CNTCV2)	32	R/W	0000_0000h	<a href="#">8.1.1.3/225</a>
2B0_0020	Counter frequency mode table base frequency register (Secure_system_counter_CNTFID0)	32	R/W	00BE_BC20h	<a href="#">8.1.1.4/225</a>
2B0_0024	Counter frequency mode table end frequency register (Secure_system_counter_CNTFID1)	32	R/W	0000_0000h	<a href="#">8.1.1.5/226</a>

### 8.1.1.1 Control register (Secure\_system\_counter\_CNTCR)

This register enables the counter.

Address: 2B0\_0000h base + 0h offset = 2B0\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### Secure\_system\_counter\_CNTCR field descriptions

Field	Description															
31–1 -	This field is reserved.															
0 EN	Enables the counter  0 System counter disabled. 1 System counter enabled.															

### 8.1.1.2 LSB of counter count value register (Secure\_system\_counter\_CNTCV1)

This register indicates the current LSB count value of the 64-bit counter. The register is writable only by secure writes. When the counter is enabled, the effect of writing the register is unpredictable.

Address: 2B0\_0000h base + 8h offset = 2B0\_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R																																				
W																																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					

#### Secure\_system\_counter\_CNTCV1 field descriptions

Field	Description																															
CNCTV1	Counter count value bits CNCTV[31:0]. The EN bit must be cleared before writing to this bits, otherwise the effect of the write is unpredictable. Writes to these registers are rare. In a system that uses security, these registers are writable only by secure writes.																															

**Secure\_system\_counter\_CNTCV1 field descriptions (continued)**

Field	Description
0	System counter disabled.
1	System counter enabled.

**8.1.1.3 MSB of counter count value register  
(Secure\_system\_counter\_CNTCV2)**

This register indicates the current MSB count value of the 64-bit counter. The register is writable only by secure writes. When the counter is enabled, the effect of writing the register is unpredictable.

Address: 2B0\_0000h base + Ch offset = 2B0\_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R																																		
W																																		

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0

**Secure\_system\_counter\_CNTCV2 field descriptions**

Field	Description
CNCTV2	Counter Count Value bits CNCTV[63:32]. The EN bit must be cleared before writing to this bits, otherwise the effect of the write is UNPREDICTABLE. Writes to these registers are rare. In a system that uses security, these registers are writable only by Secure writes.

**8.1.1.4 Counter frequency mode table base frequency register  
(Secure\_system\_counter\_CNTFID0)**

This register specifies the base frequency of the system counter. The system counter always works with SYS\_REF\_CLK/4 frequency clock. The initial value of this register is 25 MHz (100 MHz/4). The software needs to update this register initially to reflect the correct frequency based on system reference clock.

Address: 2B0\_0000h base + 20h offset = 2B0\_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
R																																			
W																																			

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 1 0 1 1 1 1 1 1 0 1 0 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

## **Secure\_system\_counter\_CNTFID0 field descriptions**

Field	Description
CNT_BASE	System counter base frequency. Holds the clock frequency of the counter. Initial value is set to 25 MHz that corresponds to 100 MHz SYSCLK frequency.

### **8.1.1.5 Counter frequency mode table end frequency register (Secure\_system\_counter\_CNTFID1)**

This register specifies the end frequency of the system counter.

Address: 2B0\_0000h base + 24h offset = 2B0\_0024h

## **Secure\_system\_counter\_CNTFID1 field descriptions**

Field	Description
END_MARKER	System counter END frequency. Indicates the end of frequency mode table.

### **8.1.2 Non-Secure system counter memory map/register definition**

The non-secure counter memory map provides register list for the non-secure world to access the counter values.

## **Non\_Secure\_SYS\_Counter memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2B1_0000	LSB of Counter Count Value (Non_Secure_SYS_Counter_CNCTV_RO1)	32	R	0000_0000h	<a href="#">8.1.2.1/227</a>
2B1_0004	MSB of counter count value register (Non_Secure_SYS_Counter_CNCTV2_RO2)	32	R	0000_0000h	<a href="#">8.1.2.2/227</a>

### 8.1.2.1 LSB of Counter Count Value (Non\_Secure\_SYS\_Counter\_CNCTV\_RO1)

This register is read-only and contains the current LSB count value of the 64-bit counter.

Address: 2B1\_0000h base + 0h offset = 2B1\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### Non\_Secure\_SYS\_Counter\_CNCTV\_RO1 field descriptions

Field	Description
CNCTV_RO1	Counter count value bits CNCTV[31:0]

### 8.1.2.2 MSB of counter count value register (Non\_Secure\_SYS\_Counter\_CNCTV2\_RO2)

The register is read-only and contains the current MSB count value of the 64-bit counter.

Address: 2B1\_0000h base + 4h offset = 2B1\_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### Non\_Secure\_SYS\_Counter\_CNCTV2\_RO2 field descriptions

Field	Description
CNCTV_RO2	Counter count value bits CNCTV[63:32]



# **Chapter 9**

## **Interconnect Fabric**

### **9.1 Introduction**

The interconnect fabric has QoS (quality of service) regulators enabled for data traffic originating from PCI express masters and qDMA master. The various options of QoS are bypass, fixed, limiter, and regulator modes. See [Figure 10-1](#) for the connectivity of interconnect fabric with PCI Express, qDMA, and CCI-400.

### **9.2 QoS generators**

The initiators such as PCI express masters and qDMA can optionally be configured with one of three types of QoS generators: fixed, limiter, and regulator. The functionality of each type of QoS generator is a superset of the previous and if configured so can be programmed under software control to behave like the subset type of generator. When software switches the mode of a QoS generator, bandwidth counters are reset.

#### **9.2.1 Fixed QoS generator**

A fixed QoS generator assigns an urgency to each packet. The urgency can be different for read and for write transactions. It can be configured so that the fixed read and write urgencies are programmable by software.

## 9.2.2 Limiter QoS generator

A limiter is an access control mechanism. It monitors the bandwidth of request packets sent by the initiators, by accumulating the data length of each packet and decrementing the accumulator at a specific rate. Read and write requests are accumulated together. The accumulator is decremented at an interval determined by the saturation parameter. Thereby, the accumulator represents the bandwidth within a certain sliding time window.

## 9.2.3 Regulator QoS generator

The regulator QoS generator modulates the priority value between two run-time programmable values whenever a run-time programmable bandwidth ceiling is exceeded. Socket transactions subject to such regulation are transmitted with a demoted level of urgency, instead of being stalled as in limiter mode.

## 9.3 IF Memory Map/Register Definition

This table shows the register memory map for the interconnect.

**IF memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2CA_2688	Priority qdma (IF_prio_qdma_read_only)	32	R/W	8000_0100h	<a href="#">9.3.1/231</a>
2CA_268C	Mode qdma (IF_mod_qdma_read_only)	32	R/W	0000_0002h	<a href="#">9.3.2/232</a>
2CA_2690	Bandwidth qdma (IF_bw_qdma_read_only)	32	R/W	0000_0005h	<a href="#">9.3.3/232</a>
2CA_2694	Saturation qdma (IF_sat_qdma_read_only)	32	R/W	0000_0100h	<a href="#">9.3.4/233</a>
2CA_2698	ExtControl qdma (IF_ext_cntrl_qdma_read_only)	32	R/W	0000_0000h	<a href="#">9.3.5/233</a>
2CA_2708	Priority qdma (IF_prio_qdma_write_only)	32	R/W	8000_0100h	<a href="#">9.3.6/234</a>
2CA_270C	Mode qdma (IF_mod_qdma_write_only)	32	R/W	0000_0002h	<a href="#">9.3.7/234</a>
2CA_2710	Bandwidth qdma (IF_bw_qdma_write_only)	32	R/W	0000_0005h	<a href="#">9.3.8/235</a>
2CA_2714	Saturation qdma (IF_sat_qdma_write_only)	32	R/W	0000_0100h	<a href="#">9.3.9/235</a>
2CA_2718	ExtControl qdma (IF_ext_cntrl_qdma_write_only)	32	R/W	0000_0000h	<a href="#">9.3.10/236</a>
2CA_2788	Priority pex (IF_prio_pex_read_only)	32	R/W	8000_0100h	<a href="#">9.3.11/236</a>
2CA_278C	Mode_pex (IF_mod_pex_read_only)	32	R/W	0000_0002h	<a href="#">9.3.12/237</a>
2CA_2790	Bandwidth_pex (IF_bw_pex_read_only)	32	R/W	0000_0005h	<a href="#">9.3.13/238</a>
2CA_2794	Saturation_pex (IF_sat_pex_read_only)	32	R/W	0000_0100h	<a href="#">9.3.14/238</a>
2CA_2798	ExtControl_pex_ro (IF_ext_cntrl_pex_ro)	32	R/W	0000_0000h	<a href="#">9.3.15/239</a>

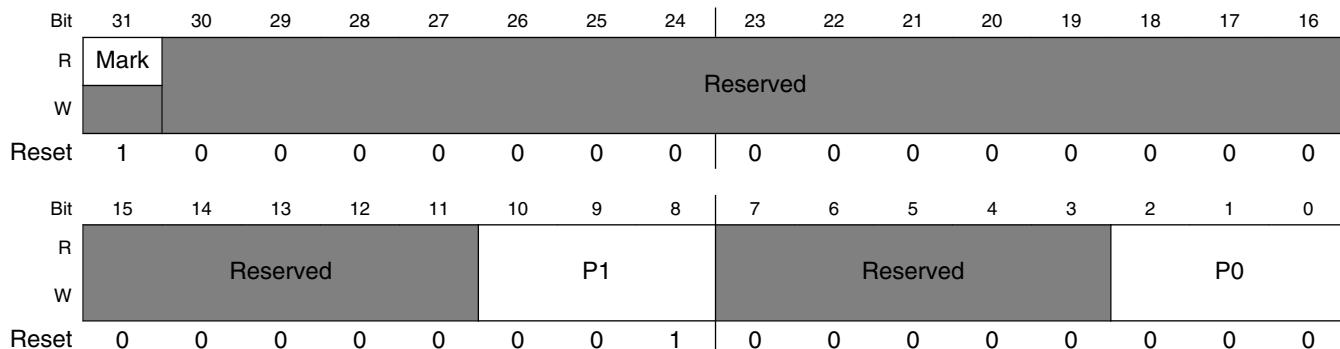
*Table continues on the next page...*

### IF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2CA_2808	Priority_pex (IF_prio_pex_write_only)	32	R/W	8000_0100h	<a href="#">9.3.16/239</a>
2CA_280C	Mode_pex (IF_mod_pex_write_only)	32	R/W	0000_0002h	<a href="#">9.3.17/240</a>
2CA_2810	Bandwidth_pex (IF_bw_smmu_pex_write_only)	32	R/W	0000_0005h	<a href="#">9.3.18/241</a>
2CA_2814	Saturation_pex (IF_sat_pex_write_only)	32	R/W	0000_0100h	<a href="#">9.3.19/241</a>
2CA_2818	ExtControl_pex_wo (IF_ext_cntrl_pex_write_only)	32	R/W	0000_0000h	<a href="#">9.3.20/242</a>

#### 9.3.1 Priority qdma (IF\_prio\_qdma\_read\_only)

Address: 2CA\_0000h base + 2688h offset = 2CA\_2688h



#### IF\_prio\_qdma\_read\_only field descriptions

Field	Description
31 Mark	It works as backward compatibility marker when set to 0.
30–11 —	This field is reserved.
10–8 P1	In programmable or bandwidth limiter mode the priority level for read transactions. In bandwidth regulator mode the priority level when the used throughput is below the threshold. In bandwidth regulator mode P1 should have a value equal or greater than P0.
7–3 —	This field is reserved.
P0	In programmable or bandwidth limiter mode the priority level for write transactions. In bandwidth regulator mode the priority level when the used throughput is above the threshold. In bandwidth regulator mode P0 should have a value equal or lower than P1.

### 9.3.2 Mode qdma (IF\_mod\_qdma\_read\_only)

Address: 2CA\_0000h base + 268Ch offset = 2CA\_268Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IF\_mod\_qdma\_read\_only field descriptions

Field	Description
31–2 —	This field is reserved.
Mode	00 Programmable mode: a programmed priority is assigned to each read or write. 01 Bandwidth limiter mode: a hard limit restricts throughput. 10 Bypass mode. 11 Bandwidth regulator mode: priority decreases when throughput exceeds a threshold.

### 9.3.3 Bandwidth qdma (IF\_bw\_qdma\_read\_only)

Address: 2CA\_0000h base + 2690h offset = 2CA\_2690h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
R																																			
W																																			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1

#### IF\_bw\_qdma\_read\_only field descriptions

Field	Description
31–13 —	This field is reserved.
Bandwidth	In bandwidth limiter or bandwidth regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example 80 Mbps on a 250 MHz interface is value 0x0052.

### 9.3.4 Saturation qdma (IF\_sat\_qdma\_read\_only)

Address: 2CA\_0000h base + 2694h offset = 2CA\_2694h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																	
W																																	

Reset 0

#### IF\_sat\_qdma\_read\_only field descriptions

Field	Description
31–10 —	This field is reserved.
Saturation	In bandwidth limiter or bandwidth regulator mode, the maximum data count value in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.

### 9.3.5 ExtControl qdma (IF\_ext\_cntrl\_qdma\_read\_only)

Address: 2CA\_0000h base + 2698h offset = 2CA\_2698h

Bit	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16															
R																																
W																																

Reset 0

Bit	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0																
R																																	
W																																	

Reset 0

#### IF\_ext\_cntrl\_qdma\_read\_only field descriptions

Field	Description
31–3 —	This field is reserved.
2 IntClkEn	IntClkEn
1 ExtThrEn	ExtThrEn
0 SocketQosEn	SocketQosEn

### 9.3.6 Priority qdma (IF\_prio\_qdma\_write\_only)

Address: 2CA\_0000h base + 2708h offset = 2CA\_2708h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Mark								Reserved							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			Reserved				P1		Reserved				P0			
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

#### IF\_prio\_qdma\_write\_only field descriptions

Field	Description
31 Mark	It works as backward compatibility marker when set to 0.
30–11 —	This field is reserved.
10–8 P1	In programmable or bandwidth limiter mode the priority level for read transactions. In bandwidth regulator mode the priority level when the used throughput is below the threshold. In bandwidth regulator mode P1 should have a value equal or greater than P0.
7–3 —	This field is reserved.
P0	In programmable or bandwidth limiter mode the priority level for write transactions. In bandwidth regulator mode the priority level when the used throughput is above the threshold. In bandwidth regulator mode P0 should have a value equal or lower than P1.

### 9.3.7 Mode qdma (IF\_mod\_qdma\_write\_only)

Address: 2CA\_0000h base + 270Ch offset = 2CA\_270Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			Reserved								Mode					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

## **IF\_mod\_qdma\_write\_only field descriptions**

Field	Description
31–2 —	This field is reserved.
Mode	00 Programmable mode: a programmed priority is assigned to each read or write. 01 Bandwidth limiter mode: a hard limit restricts throughput. 10 Bypass mode. 11 Bandwidth regulator mode: priority decreases when throughput exceeds a threshold.

### 9.3.8 Bandwidth qdma (IF\_bw\_qdma\_write\_only)

Address: 2CA 0000h base + 2710h offset = 2CA 2710h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	Reserved															Bandwidth																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1		

## **IF\_bw\_qdma\_write\_only field descriptions**

Field	Description
31–13 —	This field is reserved.
Bandwidth	In bandwidth limiter or bandwidth regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example 80 Mbps on a 250 MHz interface is value 0x0052.

### 9.3.9 Saturation qdma (IF sat qdma write only)

Address: 2CA\_0000h base + 2714h offset = 2CA\_2714h

## IF sat qdma write only field descriptions

Field	Description
31–10 —	This field is reserved.
Saturation	In bandwidth limiter or bandwidth regulator mode, the maximum data count value in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.

### 9.3.10 ExtControl qdma (IF\_ext\_cntrl\_qdma\_write\_only)

Address: 2CA\_0000h base + 2718h offset = 2CA\_2718h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												IntClkEn	ExtThrEn	SocketQosEn	
W													0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IF\_ext\_cntrl\_qdma\_write\_only field descriptions

Field	Description
31–3 —	This field is reserved.
2 IntClkEn	—
1 ExtThrEn	ExtThrEn
0 SocketQosEn	SocketQosEn

### 9.3.11 Priority pex (IF\_prio\_pex\_read\_only)

Address: 2CA\_0000h base + 2788h offset = 2CA\_2788h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Mark	Reserved														
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					P1			Reserved					P0		
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

### IF\_prio\_pex\_read\_only field descriptions

Field	Description
31 Mark	It works as backward compatibility marker when set to 0.
30–11 —	This field is reserved.
10–8 P1	In programmable or bandwidth limiter mode the priority level for read transactions. In bandwidth regulator mode the priority level when the used throughput is below the threshold. In bandwidth regulator mode P1 should have a value equal or greater than P0.
7–3 —	This field is reserved.
P0	In programmable or bandwidth limiter mode the priority level for write transactions. In bandwidth regulator mode the priority level when the used throughput is above the threshold. In bandwidth regulator mode P0 should have a value equal or lower than P1.

### 9.3.12 Mode\_pex (IF\_mod\_pex\_read\_only)

Address: 2CA\_0000h base + 278Ch offset = 2CA\_278Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															Mode
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IF\_mod\_pex\_read\_only field descriptions

Field	Description
31–2 —	This field is reserved.
Mode	00 Programmable mode: a programmed priority is assigned to each read or write. 01 Bandwidth limiter mode: a hard limit restricts throughput. 10 Bypass mode. 11 Bandwidth regulator mode: priority decreases when throughput exceeds a threshold.

### 9.3.13 Bandwidth\_pex (IF\_bw\_pex\_read\_only)

Address: 2CA 0000h base + 2790h offset = 2CA 2790h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	Reserved													Bandwidth																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1		

## **IF\_bw\_pex\_read\_only field descriptions**

Field	Description
31–13 —	This field is reserved.
Bandwidth	In bandwidth limiter or bandwidth regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example 80 Mbps on a 250 MHz interface is value 0x0052.

### 9.3.14 Saturation\_pex (IF\_sat\_pex\_read\_only)

Address: 2CA 0000h base + 2794h offset = 2CA 2794h

## **IF\_sat\_pex\_read\_only field descriptions**

Field	Description
31–10 —	This field is reserved.
Saturation	In bandwidth limiter or bandwidth regulator mode, the maximum data count value in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.

### 9.3.15 ExtControl\_pex\_ro (IF\_ext\_cntrl\_pex\_ro)

Address: 2CA\_0000h base + 2798h offset = 2CA\_2798h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												IntClkEn	ExtThrEn	SocketQosEn	
W													0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IF\_ext\_cntrl\_pex\_ro field descriptions

Field	Description
31–3 —	This field is reserved.
2 IntClkEn	IntClkEn
1 ExtThrEn	ExtThrEn
0 SocketQosEn	SocketQosEn

### 9.3.16 Priority\_pex (IF\_prio\_pex\_write\_only)

Address: 2CA\_0000h base + 2808h offset = 2CA\_2808h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Mark		Reserved													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					P1			Reserved					P0		
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

**IF\_prio\_pex\_write\_only field descriptions**

Field	Description
31 Mark	It works as backward compatibility marker when set to 0.
30–11 —	This field is reserved.
10–8 P1	In programmable or bandwidth limiter mode the priority level for read transactions. In bandwidth regulator mode the priority level when the used throughput is below the threshold. In bandwidth regulator mode P1 should have a value equal or greater than P0.
7–3 —	This field is reserved.
P0	In programmable or bandwidth limiter mode the priority level for write transactions. In bandwidth regulator mode the priority level when the used throughput is above the threshold. In bandwidth regulator mode P0 should have a value equal or lower than P1.

**9.3.17 Mode\_pex (IF\_mod\_pex\_write\_only)**

Address: 2CA\_0000h base + 280Ch offset = 2CA\_280Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															Mode
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IF\_mod\_pex\_write\_only field descriptions**

Field	Description
31–2 —	This field is reserved.
Mode	00 Programmable mode: a programmed priority is assigned to each read or write. 01 Bandwidth limiter mode: a hard limit restricts throughput. 10 Bypass mode. 11 Bandwidth regulator mode: priority decreases when throughput exceeds a threshold.

### 9.3.18 Bandwidth\_pex (IF\_bw\_smmu\_pex\_write\_only)

Address: 2CA\_0000h base + 2810h offset = 2CA\_2810h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										Bandwidth																					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1		

## **IF\_bw\_smmu\_pex\_write\_only field descriptions**

Field	Description
31–13 —	This field is reserved.
Bandwidth	In bandwidth limiter or bandwidth regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example 80 Mbps on a 250 MHz interface is value 0x0052.

### 9.3.19 Saturation\_pex (IF\_sat\_pex\_write\_only)

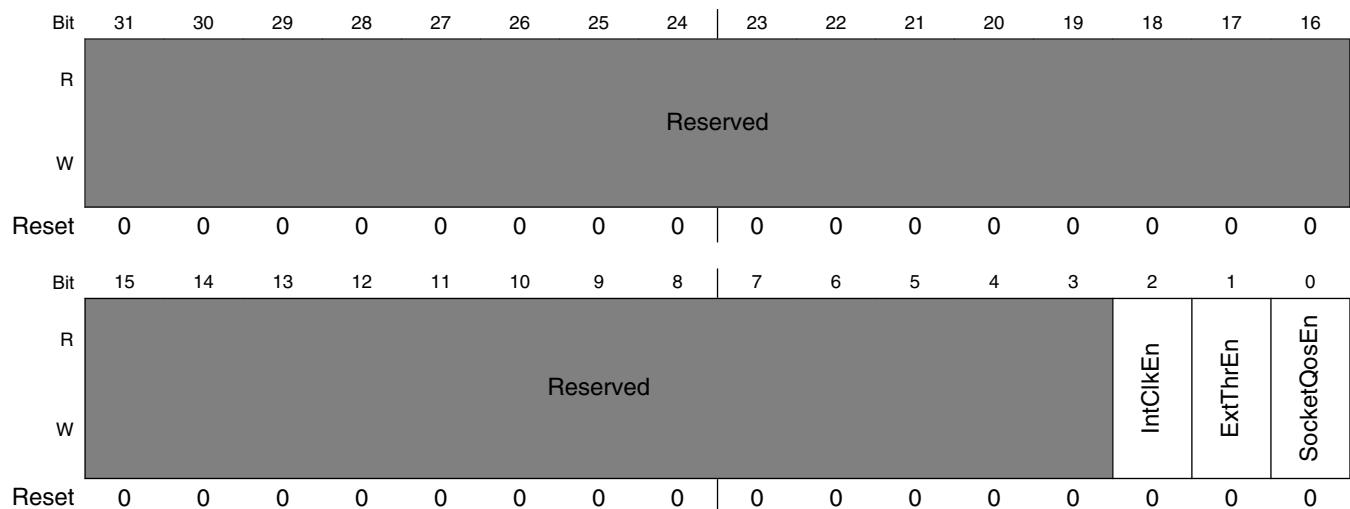
Address: 2CA\_0000h base + 2814h offset = 2CA\_2814h

## **IF\_sat\_pex\_write\_only field descriptions**

Field	Description
31–10 —	This field is reserved.
Saturation	In bandwidth limiter or bandwidth regulator mode, the maximum data count value in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.

### 9.3.20 ExtControl\_pex\_wo (IF\_ext\_cntrl\_pex\_write\_only)

Address: 2CA\_0000h base + 2818h offset = 2CA\_2818h



**IF\_ext\_cntrl\_pex\_write\_only field descriptions**

Field	Description
31–3 —	This field is reserved.
2 IntClkEn	IntClkEn
1 ExtThrEn	ExtThrEn
0 SocketQosEn	SocketQosEn

# Chapter 10

## Cache Coherent Interconnect (CCI-400)

### 10.1 The CCI-400 module as implemented on the chip

The CCI-400 is the main coherent interconnect of the chip. The CCI-400 interconnect supports two ACE masters, three-Ace-Lite masters, and three slave interfaces.

This section provides details about how the CCI-400 module is implemented on the chip.

#### 10.1.1 LS1043A CCI module integration

The register offset in CCI-400 starts from 0x90000 and should be treated as 0x0000 due to implementation. The CCI-400 base address is 0x1180000 for LS1043A. See [Figure 10-1](#) below for the MMU-500 and CCI-400 connectivity.

The CCI-400 supports coherent/snoop transactions (write unique and read once) from the masters. The master that doesn't have the capability to generate coherent/snoop transactions have chip defined registers ([Snoop Configuration Register \(SCFG\\_SNPCNFGCR\)](#)) to generate snoop transactions. Once the corresponding field in this register is set, all the transactions are considered as snoop/coherent data.

Unlike the other masters where certain data transactions can be coherent/snoop, the masters whose snoop is enabled through this register, all transactions are considered snoop/coherent.

Similarly there exists QOS ([QOS1 Register \(SCFG\\_QOS1\)](#), [QOS2 Register \(SCFG\\_QOS2\)](#)) for programming QoS for data transactions from the corresponding masters. The masters which can generate QoS are not mentioned in this register.

Only nERRORIRQ interrupt is connected to the interrupt controller.

#### NOTE

The CCI-400 has per core lock-reservation monitor individually for secure and non-secure exclusive (ARLOCK/AWLOCK)

transactions (that is, a total of eight monitors). These transactions are terminated at the CCI level, as the system beyond CCI do not support the “lock transactions”. Transaction should be marked with domain setting, such as “01- Inner sharable” or “10- Outershareable”.

## 10.1.2 Connections to the CCI-400 interconnect

The following are the connections to the CCI-400 interconnect:

- 2x full ACE slave ports. One (S4 128-b ACE interface) is used to connect the quad core A53 core platform. The other (S3) is left unused.
- 3x ACE-Lite and DVM ports, used to connect to IO-Coherent masters
  - S0 Ace-Lite slave interface (FMan, SEC, QMan/BMan masters through NoC and SMMU-500 TCU and TBU)
  - S1 Ace-Lite slave interface (qDMA and PCI express masters through Interconnect fabric and TBU1, TBU2)
  - S2 Ace-Lite slave interface (All other masters through Interconnect Fabric and SMMU-500 TBU3)

Three ACE-Lite master ports

- M0 for peripheral access (all the peripherals except PCI express controllers through Interconnect fabric)
- M1 AXI3 allocated for DDR memory
- M2 (connected to Interconnect Fabric through CPE425 for PCI express traffic)

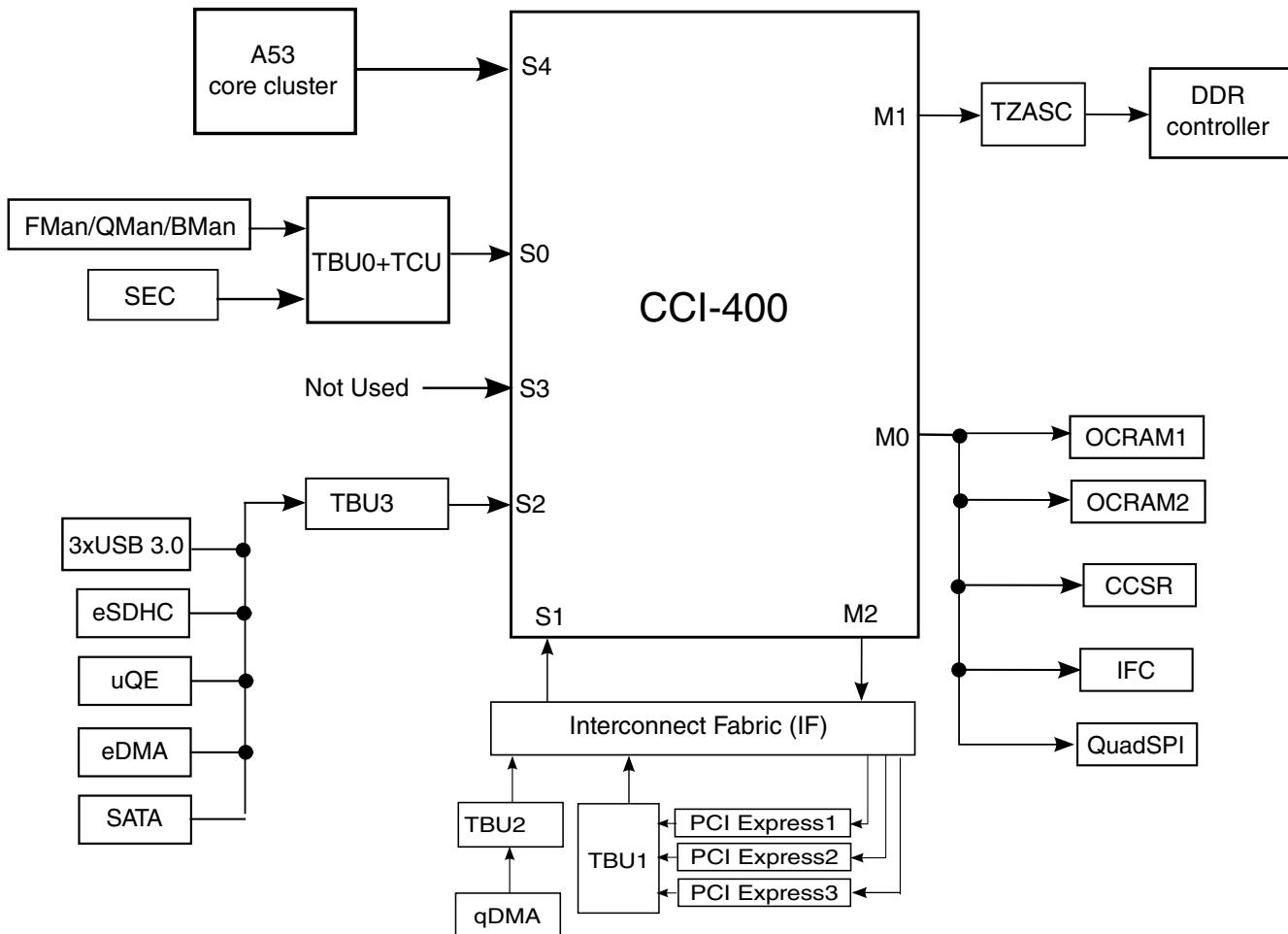


Figure 10-1. Chip interconnect

### 10.1.3 Snoop transaction configurations

The following table provides the snoop transaction configurations for different modules:

Table 10-1. Snoop transaction configuration for modules

Module	Snoop transaction configuration
FMAN	Refer LS1043ADPAARM for more details.
BMAN/QMAN	
SEC	All transactions from SEC are tagged as snoop configuration if the SCFG_SNPCNFGCR[SECRDSNP], SCFG_SNPCNFGCR[SECWRSNP], SEC_MCFG[ARCACHE], and MCFG[AWCACHE] bits are set.

*Table continues on the next page...*

**Table 10-1. Snoop transaction configuration for modules (continued)**

Module	Snoop transaction configuration
	Refer <a href="#">Snoop Configuration Register (SCFG_SNPCNFGCR)</a> and Master Configuration (MCFGR) in security reference manual for details.
PCI Expressn	Snoop transactions are dependent on PCI Express protocol and controlled by PCI Express IATU registers. Refer <a href="#">PEX register descriptions</a> for details.
qDMA	Global snoop is controlled by DMA mode register, legacy mode - DMA legacy attribute register, queue mode - command queue mode register. Refer <a href="#">DMA mode register (DMR)</a> , <a href="#">Command queue mode register (CQMR)</a> for details.
SATA	All transactions from SATA are tagged as snoop configuration if the SCFG_SNPCNFGCR[SATARDSNP], SCFG_SNPCNFGCR[SATAWRSNP], SATA_AXICC[EARC], and SATA_AXICC[EAWC] bits are set. Refer <a href="#">Snoop Configuration Register (SCFG_SNPCNFGCR)</a> and <a href="#">AXI cache control register (AXICC)</a> for details.
USBn3.0	All transactions from USB 3.0 are tagged as snoop configuration if the SCFG_SNPCNFGCR[USBnRDSNP], SCFG_SNPCNFGCR[USBnWRSNP] and USBx_GSBUSCFG0 bits are set. Refer <a href="#">Snoop Configuration Register (SCFG_SNPCNFGCR)</a> and <a href="#">Global SoC Bus Configuration Register 0 (USB_GSBUSCFG0)</a> for details.
QUICC Engine	Snoop transactions from QUICC Engine are controlled by bus mode register. Refer to the bus mode register in QEWRM for details.
eDMA	All transactions from eDMA are tagged as snoop configuration if the SCFG_SNPCNFGCR[eDMASNP] bit is set. Refer <a href="#">Snoop Configuration Register (SCFG_SNPCNFGCR)</a> for details.

## 10.2 CCI400 register descriptions

### 10.2.1 CCI400 Registers memory map

CCI400 base address: 118\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	<a href="#">Control Override Register (Control_OVERRIDE_Register)</a>	32	RW	<a href="#">Table 10-1</a>
4h	<a href="#">Speculation Control Register (Speculation_Control_Register)</a>	32	RW	<a href="#">Table 10-1</a>
8h	<a href="#">Secure Access Register (Secure_Access_Register)</a>	32	RW	<a href="#">Table 10-1</a>
Ch	<a href="#">Status Register (Status_Register)</a>	32	RO	<a href="#">Table 10-1</a>
10h	<a href="#">Imprecise Error Register (Imprecise_Error_Register)</a>	32	RW	<a href="#">Table 10-1</a>
1000h	<a href="#">Snoop Control Registers (Snoop_Control_Register_S0)</a>	32	RW	<a href="#">Table 10-1</a>
1004h	<a href="#">Shareable Override Registers (Shareable_Override_Register_S0)</a>	32	RW	<a href="#">Table 10-1</a>
1100h	<a href="#">Read Channel QoS Value Override Register (Read_Qos_Override_Register_S0)</a>	32	RW	<a href="#">Table 10-1</a>
1104h	<a href="#">Write Qos Override Register (Write_Qos_Override_Register_S0)</a>	32	RW	<a href="#">Table 10-1</a>
110Ch	<a href="#">Qos Control Register (Qos_Control_Register_S0)</a>	32	RW	<a href="#">Table 10-1</a>

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1110h	Max OT Registers (Max_OT_Register_S0)	32	RW	Table 10-1
1130h	Regulator Target Registers (Target_Latency_Register_S0)	32	RW	Table 10-1
1138h	QoS Range Register (Qos_Range_Register_S0)	32	RW	Table 10-1
2000h	Snoop Control Registers (Snoop_Control_Register_S1)	32	RW	Table 10-1
2004h	Shareable Override Registers (Shareable_Override_Register_S1)	32	RW	Table 10-1
2100h	Read Channel QoS Value Override Register (Read_Qos_Override_Register_S1)	32	RW	Table 10-1
2104h	Write Qos Override Register (Write_Qos_Override_Register_S1)	32	RW	Table 10-1
210Ch	Qos Control Register (Qos_Control_Register_S1)	32	RW	Table 10-1
2110h	Max OT Registers (Max_OT_Register_S1)	32	RW	Table 10-1
2130h	Regulator Target Registers (Target_Latency_Register_S1)	32	RW	Table 10-1
2138h	QoS Range Register (Qos_Range_Register_S1)	32	RW	Table 10-1
2268h	QoS Regulator Scale Factor Registers (Latency_Regulation_Regis ter_S0)	32	RW	Table 10-2
3000h	Snoop Control Registers (Snoop_Control_Register_S2)	32	RW	Table 10-1
3004h	Shareable Override Registers (Shareable_Override_Register_S2)	32	RW	Table 10-1
3100h	Read Channel QoS Value Override Register (Read_Qos_Override_Register_S2)	32	RW	Table 10-1
3104h	Write Qos Override Register (Write_Qos_Override_Register_S2)	32	RW	Table 10-1
310Ch	Qos Control Register (Qos_Control_Register_S2)	32	RW	Table 10-1
3110h	Max OT Registers (Max_OT_Register_S2)	32	RW	Table 10-1
3130h	Regulator Target Registers (Target_Latency_Register_S2)	32	RW	Table 10-1
3138h	QoS Range Register (Qos_Range_Register_S2)	32	RW	Table 10-1
3268h	QoS Regulator Scale Factor Registers (Latency_Regulation_Regis ter_S1)	32	RW	Table 10-2
4000h	Snoop Control Registers (Snoop_Control_Register_S3)	32	RW	Table 10-1
4004h	Shareable Override Registers (Shareable_Override_Register_S3)	32	RW	Table 10-1
4100h	Read Channel QoS Value Override Register (Read_Qos_Override_Register_S3)	32	RW	Table 10-1
4104h	Write Qos Override Register (Write_Qos_Override_Register_S3)	32	RW	Table 10-1
410Ch	Qos Control Register (Qos_Control_Register_S3)	32	RW	Table 10-1
4110h	Max OT Registers (Max_OT_Register_S3)	32	RW	Table 10-1
4130h	Regulator Target Registers (Target_Latency_Register_S3)	32	RW	Table 10-1
4138h	QoS Range Register (Qos_Range_Register_S3)	32	RW	Table 10-1
4268h	QoS Regulator Scale Factor Registers (Latency_Regulation_Regis ter_S2)	32	RW	Table 10-2
5000h	Snoop Control Registers (Snoop_Control_Register_S4)	32	RW	Table 10-1
5004h	Shareable Override Registers (Shareable_Override_Register_S4)	32	RW	Table 10-1
5100h	Read Channel QoS Value Override Register (Read_Qos_Override_Register_S4)	32	RW	Table 10-1
5104h	Write Qos Override Register (Write_Qos_Override_Register_S4)	32	RW	Table 10-1
510Ch	Qos Control Register (Qos_Control_Register_S4)	32	RW	Table 10-1

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
5110h	Max OT Registers (Max_OT_Register_S4)	32	RW	<a href="#">Table 10-1</a>
5130h	Regulator Target Registers (Target_Latency_Register_S4)	32	RW	<a href="#">Table 10-1</a>
5138h	QoS Range Register (Qos_Range_Register_S4)	32	RW	<a href="#">Table 10-1</a>
5268h	QoS Regulator Scale Factor Registers (Latency_Regulation_Regis ter_S3)	32	RW	<a href="#">Table 10-2</a>
6268h	QoS Regulator Scale Factor Registers (Latency_Regulation_Regis ter_S4)	32	RW	<a href="#">Table 10-2</a>

## 10.2.2 Control Override Register (Control\_Override\_Register)

### 10.2.2.1 Offset

Register	Offset
Control_Override_Reg ister	0h

### 10.2.2.2 Function

The Control Override register is an additional control register that provides a fail-safe override for some CCI-400 functions, if these cause problems that you cannot otherwise work around. If you cannot avoid using them, only set them using non-bufferable transactions, and before barriers, shareable transactions, or DVM messages are issued into the CCI-400. This can be, for example, very early in the boot sequence, prior to the installation of any Secure OS. You can access the Control Override Register using Secure transactions only, irrespective of the programming of the Secure Access Register . Available in all CCI-400 configurations.

### 10.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W	Reserved																
Reset	u	u	u	u	u	u	u	u	u	u	u	0	0	0	0	0	0

### 10.2.2.4 Fields

Field	Function
31-6 —	Reserved
5 Disable_Retry_Reduction_Buffers	Disable retry reduction buffers for speculative fetches 0b - Retry reduction buffers enabled. 1b - Retry reduction buffers disabled.
4 Disable_Priority_Promotion	ARQOSARBS inputs are ignored 0b - The CCI-400 uses ARQOSARBS inputs to promote the priority of earlier requests. 1b - The CCI-400 ignores ARQOSARBS inputs.
3 Terminate_BARRIERS	Terminate Barriers 0b - Master interfaces terminate barriers according to the BARRIERTERMINATE inputs. 1b - All master interfaces terminate barriers.
2 Disable_Speculative_Fetches	Disable Speculative Fetches 0b - Send speculative fetches according to the Speculation Control Register. See Speculation Control Register. 1b - Disable speculative fetches from all master interfaces.
1	DVM Message Disable 0b - Send DVM messages according to the Snoop Control Registers. See Snoop Control Registers.

Table continues on the next page...

## CCI400 register descriptions

Field	Function
DVM_Message_Disable	1b - Disable propagation of all DVM messages.
0 Snoop_Disable	Snoop Disable 0b - Snoop masters according to the Snoop Control Registers. See Snoop Control Registers. 1b - Disable all snoops, but not DVM messages.

## 10.2.3 Speculation Control Register (Speculation\_Control\_Register)

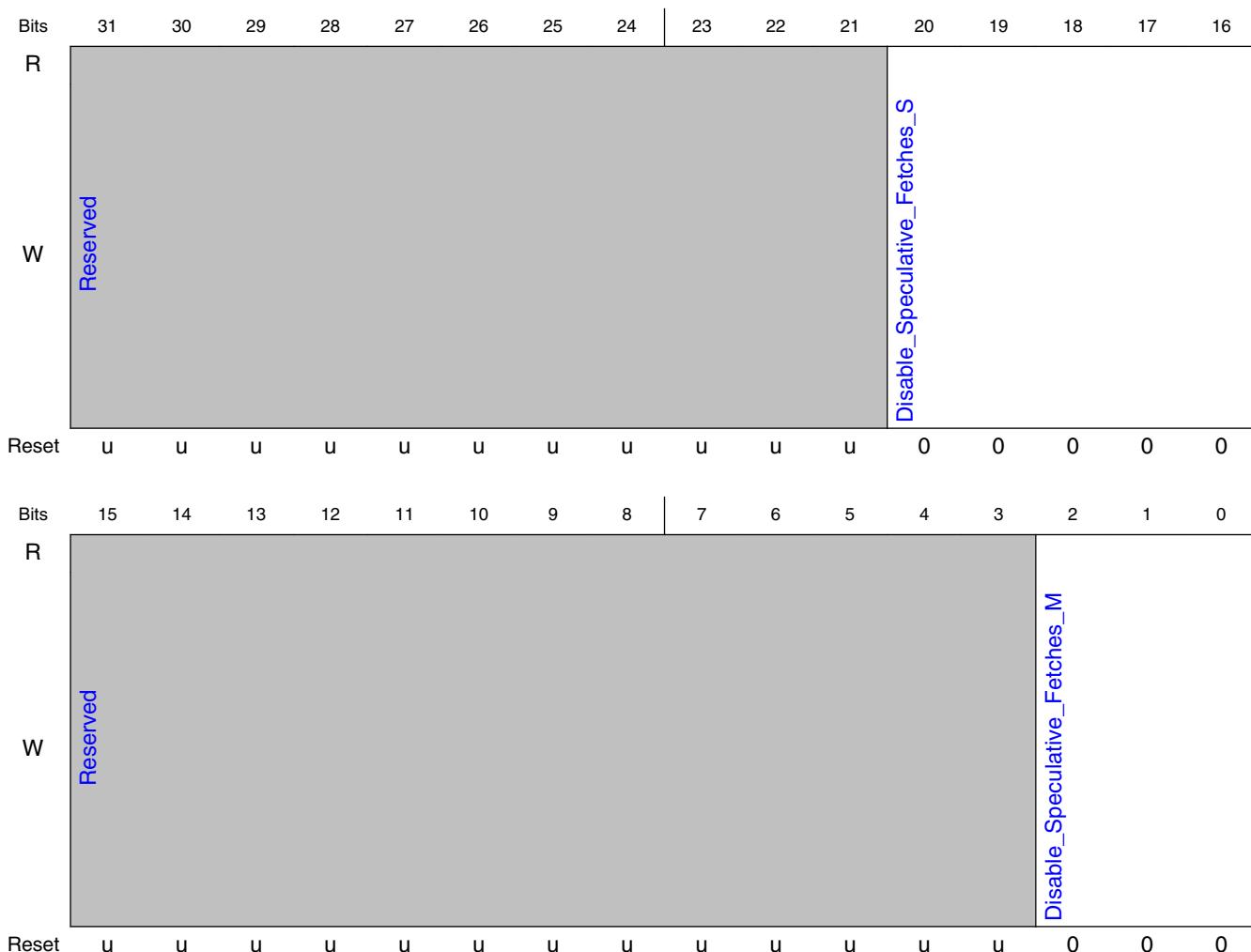
### 10.2.3.1 Offset

Register	Offset
Speculation_Control_Register	4h

### 10.2.3.2 Function

The Speculation Control register disables speculative fetches for a master interface or for traffic through a specific slave interface. Speculative fetches are not issued if they are disabled in either the slave or master interface for a particular transaction. Access controlled by Secure Access Register, see Secure Access Register. Available in all CCI-400 configurations.

### 10.2.3.3 Diagram



### 10.2.3.4 Fields

Field	Function
31-21 —	Reserved
20-16 Disable_Speculative_Fetches_S	Disable speculative fetches from slave Disable speculative fetches for transactions through a slave interface. One bit for each slave interface: S4, S3, S2, S1, and S0: 00000b - Enable speculative fetches. 00001b - Disable speculative fetches.
15-3 —	Reserved

Table continues on the next page...

Field	Function
2-0	Disable speculative fetches from master
Disable_Speculative_Fetches_M	Disable speculative fetches from a master interface. One bit for each master interface: M2, M1, and M0. 000b - Enable speculative fetches. 001b - Disable speculative fetches.

## 10.2.4 Secure Access Register (Secure\_Access\_Register)

### 10.2.4.1 Offset

Register	Offset
Secure_Access_Register	8h

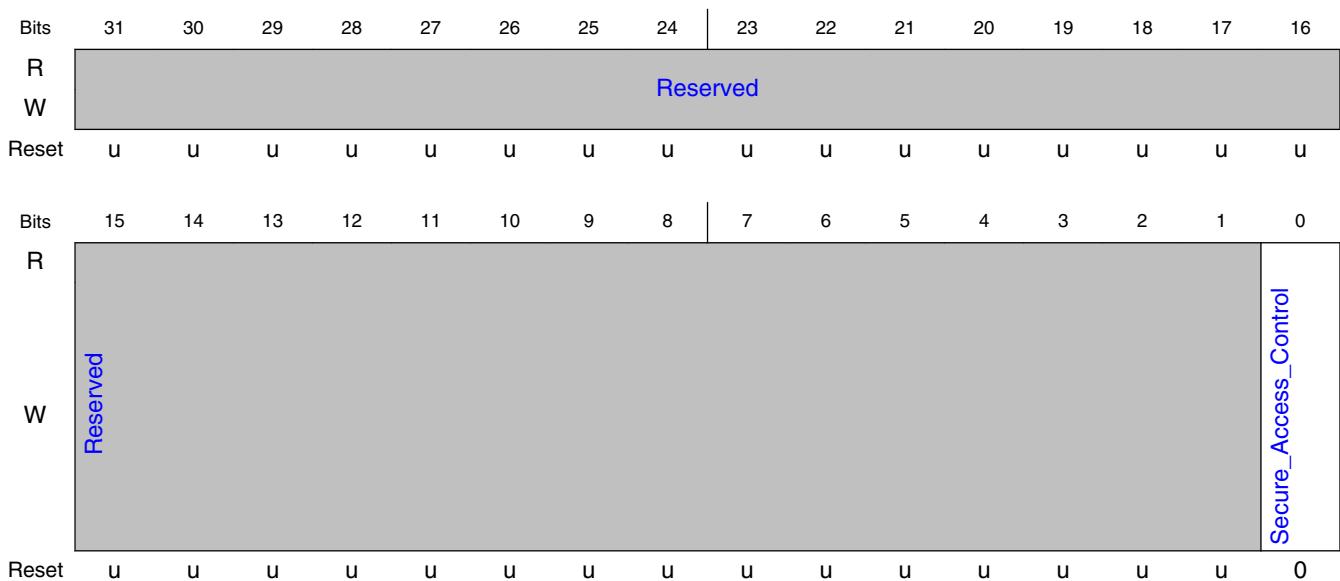
### 10.2.4.2 Function

The Secure Access register controls secure access. You can only write to this register using Secure transactions. Available in all CCI-400 configurations.

#### NOTE

This register enables Non-secure access to the CCI-400 registers for all masters. This compromises the security of your system.

### **10.2.4.3 Diagram**



#### **10.2.4.4 Fields**

Field	Function
31-1 —	Reserved
0 Secure_Access_Control	Secure Access Control Non-secure register access override: 0b - Disable Non-secure access to CCI-400 registers. 1b - Enable Non-secure access to CCI-400 registers.

### **10.2.5 Status Register (Status\_Register)**

### 10.2.5.1 Offset

<b>Register</b>	<b>Offset</b>
Status_Register	Ch

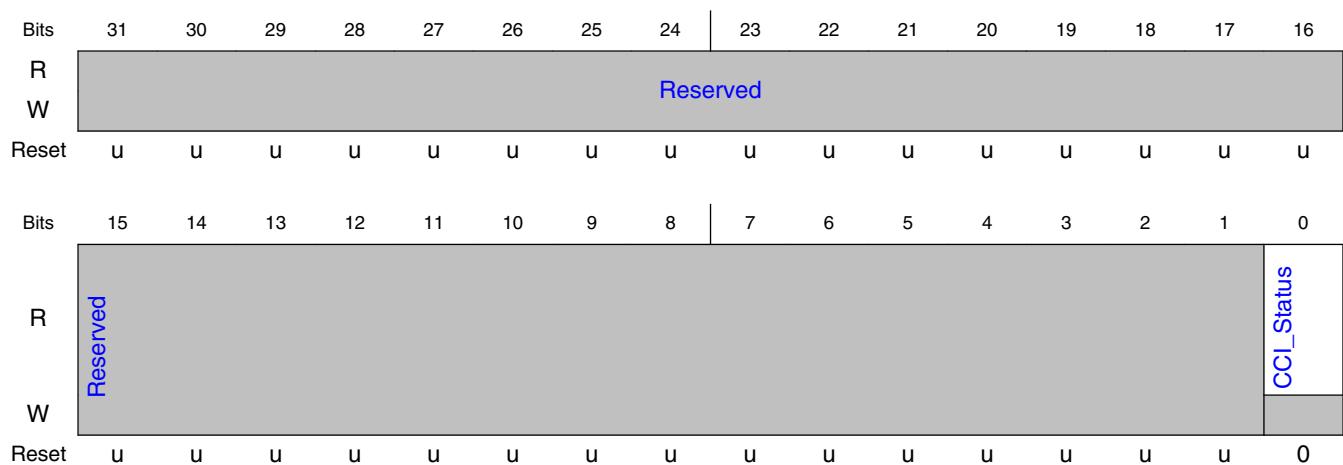
### 10.2.5.2 Function

The Status Register safely enables and disables snooping. When changing the snoop or DVM message enables using the Snoop Control Registers, see Snoop Control Registers, there is a delay until the changes are registered in all parts of the CCI-400. The change\_pending bit in the Status Register indicates whether there are any changes to the enables that have not yet been applied, or whether a slave interface has been disabled for future snoop and DVM messages, but has outstanding AC requests. There are no usage constraints. Available in all CCI-400 configurations.

#### NOTE

After writing to the snoop or DVM enable bits, the controller must wait for the register write to complete, then test that the change\_pending bit is LOW before it turns an attached device on or off.

### 10.2.5.3 Diagram



### 10.2.5.4 Fields

Field	Function
31-1 —	Reserved
0 CCI_Status	CCI_Status Indicates whether any changes to the snoop or DVM enables is pending in the CCI-400 0b - No change pending. 1b - Change pending.

## 10.2.6 Imprecise Error Register (Imprecise\_Error\_Register)

### 10.2.6.1 Offset

Register	Offset
Imprecise_Error_Register	10h

### 10.2.6.2 Function

The Imprecise Error register records the CCI-400 interfaces that received an error that is not signaled precisely. The appropriate bit is set, with respect to the interface on which the error was received. Bits are set when one or more error responses are detected, and they are reset on a write of 1 to the corresponding bit. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments. There are no usage constraints. Available in all CCI-400 configurations.

#### NOTE

If any of the imprecise error indicator bits are set, the nERRORIRQ signal is asserted, active LOW.

### 10.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	Reserved															
Reset	u	u	u	u	u	u	u	u	u	u	u	0	0	0	0	0

### 10.2.6.4 Fields

Field	Function
31-21 —	Reserved
20 Imp_Err_S4	Imprecise error indicator for slave interface S4
19 Imp_Err_S3	Imprecise error indicator for slave interface S3
18 Imp_Err_S2	Imprecise error indicator for slave interface S2
17 Imp_Err_S1	Imprecise error indicator for slave interface S1
16 Imp_Err_S0	Imprecise error indicator for slave interface S0
15-3 —	Reserved
2 Imp_Err_M2	Imprecise error indicator for master interface M2
1 Imp_Err_M1	Imprecise error indicator for master interface M1
0 Imp_Err_M0	Imprecise error indicator for master interface M0 0b - No error from the time this bit was last reset. 1b - An error response has been received, but not signalled precisely.

### 10.2.7 Snoop Control Registers (Snoop\_Control\_Register\_S0 - Snoop\_Control\_Register\_S4)

#### 10.2.7.1 Offset

Register	Offset
Snoop_Control_Register_S0	1000h
Snoop_Control_Register_S1	2000h
Snoop_Control_Register_S2	3000h

Table continues on the next page...

Register	Offset
Snoop_Control_Register_S3	4000h
Snoop_Control_Register_S4	5000h

### 10.2.7.2 Function

The Snoop Control register controls the issuing of snoop and DVM requests on each slave interface. You can read the register to determine if the interface supports snoops or DVM messages. Enabling snoop or DVM requests on an interface that does not support them has no effect. One Snoop Control Register exists for each slave interface.

Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments. Available in all CCI-400 configurations.

#### NOTE

- If the ACCCHANNELEN input is LOW for this interface, write accesses to this register are ignored and snoop or DVM requests cannot be enabled.
- If snoops are disabled in the Control Override Register, write accesses to the snoop enable bit[0] are ignored.
- If DVM messages are disabled in the Control Override Register, write accesses to the DVM enable bit[1] are ignored.

### 10.2.7.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Support_DVMs	Support_snoops	Reserved														
W																	
Reset	0	0	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved														Enable_DVMs	Enable_Snoop	
W															0	u	
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0	u

### 10.2.7.4 Fields

Field	Function
31 Support_DVMs	Slave interface supports DVM messages This is overridden to 0x0 if you set the Control Override Register [1]. See Control Override Register.
30 Support_snoops	Slave interface supports snoops This is overridden to 0x0 if you set the Control Override Register [0]. See Control Override Register.
29-2 —	Reserved
1 Enable_DVMs	Enable DVMs Enable issuing of DVM message requests from slave interface. RAZ/WI for interfaces not supporting DVM messages: 0b - Disable DVM message requests. 1b - Enable DVM message requests.
0 Enable_Snoop	Enable Snoop Enable issuing of snoop requests from this slave interface. RAZ/WI for interfaces not supporting snoops: <b>NOTE:</b> This bit is reserved for Snoop_Control_Register_S0, Snoop_Control_Register_S1, Snoop_Control_Register_S2, and Snoop_Control_Register_S3. 0b - Disable snoop requests. 1b - Enable snoop requests.

## 10.2.8 Shareable Override Registers (Shareable\_OVERRIDE\_Register\_S0 - Shareable\_OVERRIDE\_Register\_S4)

### 10.2.8.1 Offset

Register	Offset
Shareable_OVERRIDE_Register_S0	1004h
Shareable_OVERRIDE_Register_S1	2004h
Shareable_OVERRIDE_Register_S2	3004h
Shareable_OVERRIDE_Register_S3	4004h
Shareable_OVERRIDE_Register_S4	5004h

### 10.2.8.2 Function

The Shareable Override register overrides shareability of normal transactions through this interface. The following transaction types are unaffected by any override:

- FIXED-type bursts.
- Device transactions.
- Barrier.
- DVM message transactions.

Usage constraints This register is for ACE-Lite slave interfaces only. See the AMBA AXI and ACE Protocol Specification. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments. Available in all CCI-400 configurations.

#### NOTE

Exclusive accesses must not be issued on an interface that is being overridden as shareable. If the CCI-400 is programmed to override transactions as shareable, Exclusive accesses are overridden to normal accesses. An exclusive write then receives an OKAY response to indicate that the slave does not support exclusive accesses.

### 10.2.8.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	u	u	u	u	u	u	u	u		u	u	u	u	u	u	u	u
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W	Reserved																
Reset	u	u	u	u	u	u	u	u		u	u	u	u	u	u	0	0

### 10.2.8.4 Fields

Field	Function
31-2	Reserved
—	
1-0 AxDomain_Override	AxDOMAIN override Shareable override for slave interface 00b - Do not override AxDOMAIN inputs. 01b - Do not override AxDOMAIN inputs. 10b - Override AxDOMAIN inputs to 0b00, all transactions are treated as non-shareable. ReadOnce becomes ReadNoSnoop. WriteUnique and WriteLineUnique become WriteNoSnoop. 11b - Override AxDOMAIN inputs to 0b01, normal transactions are treated as shareable. ReadNoSnoop becomes ReadOnce. WriteNoSnoop becomes WriteUnique.

### 10.2.9 Read Channel QoS Value Override Register (Read\_Qos\_Override\_Register\_S0 - Read\_Qos\_Override\_Register\_S4)

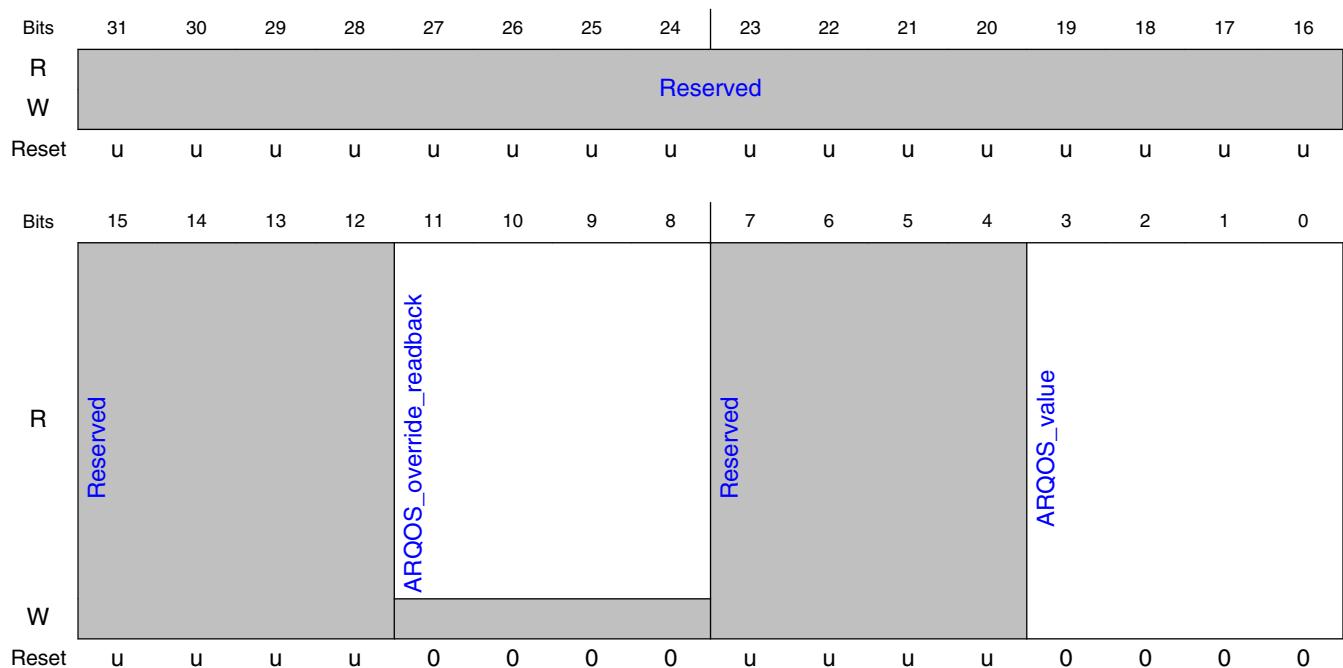
### 10.2.9.1 Offset

Register	Offset
Read_Qos_Override_Register_S0	1100h
Read_Qos_Override_Register_S1	2100h
Read_Qos_Override_Register_S2	3100h
Read_Qos_Override_Register_S3	4100h
Read_Qos_Override_Register_S4	5100h

### 10.2.9.2 Function

The Read Channel QoS Value Override register contains override values for ARQOS, with a register for each slave interface. This value is used if you set the QOS OVERRIDE[4:0] input signal bit for this slave interface and the QoS value regulator is not enabled. You can also use this register to read the current value of the QoS value regulator for read accesses when the regulator is enabled. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments. Available in all CCI-400 configurations.

### 10.2.9.3 Diagram



### 10.2.9.4 Fields

Field	Function
31-12 —	Reserved
11-8 ARQOS_overrid e_readback	ARQOS override readback Reads what value is currently applied to transactions with ARQOS=0, provided QOSOVERRIDE is HIGH and the QoS value regulator is enabled.
7-4 —	Reserved
3-0 ARQOS_value	ARQOS value ARQOS value override for slave interface

## 10.2.10 Write Qos Override Register (Write\_Qos\_Override\_Regis ter\_S0 - Write\_Qos\_Override\_Register\_S4)

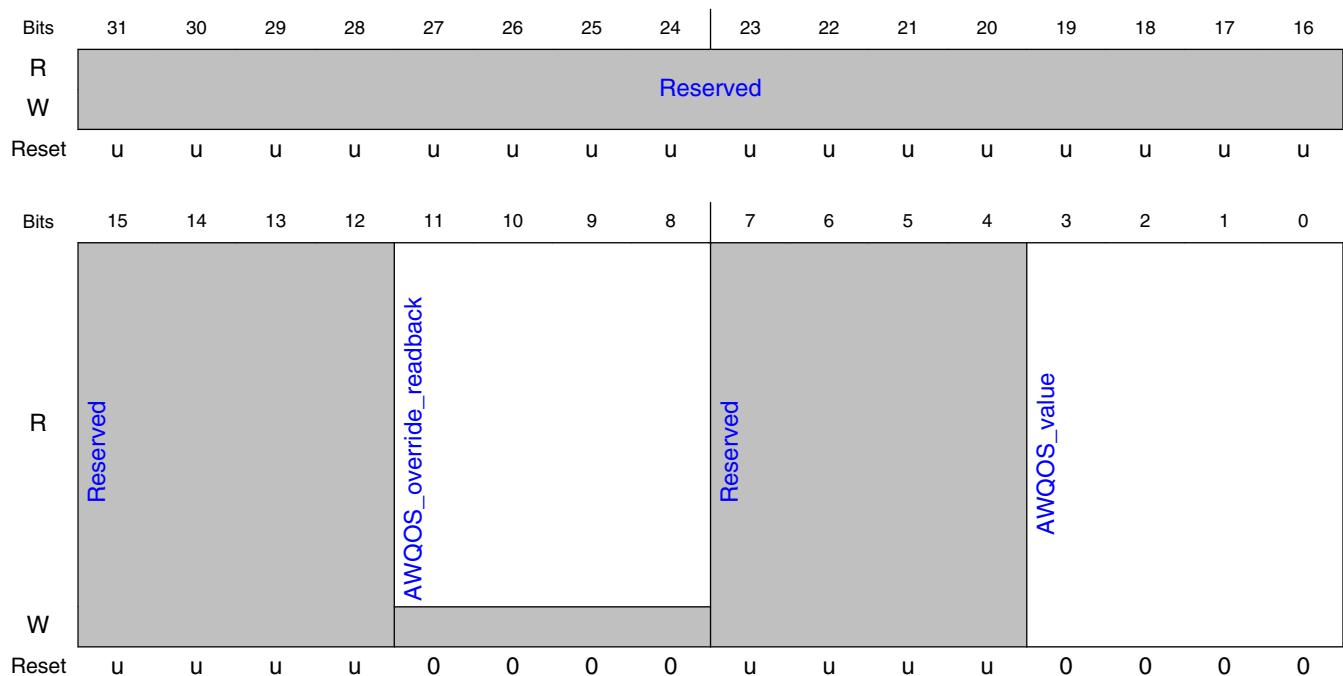
### 10.2.10.1 Offset

Register	Offset
Write_Qos_Override_Register_S0	1104h
Write_Qos_Override_Register_S1	2104h
Write_Qos_Override_Register_S2	3104h
Write_Qos_Override_Register_S3	4104h
Write_Qos_Override_Register_S4	5104h

### 10.2.10.2 Function

The Write Channel QoS Value Override Register characteristics are: Purpose Contains override values for AWQOS, with a register for each slave interface. This value is used if you set the QOSOVERRIDE[4:0] input signal bit for this slave interface and the QoS value regulator is not enabled. You can also read the current value of the QoS value regulator for write accesses when the regulator is enabled. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments. Available in all CCI-400 configurations.

### 10.2.10.3 Diagram



### 10.2.10.4 Fields

Field	Function
31-12 —	Reserved
11-8 AWQOS_overrid e_readback	AWQOS override readback Reads what value is currently applied to transactions with AWQOS=0, provided QOSOVERRIDE is HIGH and the QoS value regulator is enabled.
7-4 —	Reserved
3-0 AWQOS_value	AWQOS value AWQOS value override for slave interface S0

## 10.2.11 Qos Control Register (Qos\_Control\_Register\_S0 - Qos\_Control\_Register\_S4)

### 10.2.11.1 Offset

Register	Offset
Qos_Control_Register_S0	110Ch
Qos_Control_Register_S1	210Ch
Qos_Control_Register_S2	310Ch
Qos_Control_Register_S3	410Ch
Qos_Control_Register_S4	510Ch

### 10.2.11.2 Function

The QoS Control register controls the regulators that are enabled on the slave interfaces. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments on page 3-10. Available in all CCI-400 configurations.

#### NOTE

When outstanding transaction regulation is enabled or disabled for an interface, changes take effect only when there are no outstanding transactions in that interface.

### 10.2.11.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R																	
W																	
Reset	0	u	u	u	u	u	u	u		u	u	0	0	u	u	u	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W																	
Reset	u	u	u	u	u	u	u	u		u	u	u	0	0	0	0	0

### 10.2.11.4 Fields

Field	Function
31 QoS_regression_disabled	QoS regulation disabled Determines whether this CCI-400 implementation supports QoS regulation. 0b - QoS regulation fully supported as described in this document. See Quality of Service. 1b - QoS regulation not supported, reads and writes to this register have no effect.
30-22 —	Reserved
21 Bandwidth_regulation_mode	Bandwidth regulation mode Sets the mode for bandwidth regulation: 0b - Normal mode. The QoS value is stable when the master is idle. 1b - Quiesce High mode. The QoS value tends to the maximum when the master is idle.
20	ARQOS regulation mode Configures the mode of the QoS value regulator for read transactions:

*Table continues on the next page...*

Field	Function
ARQOS_regulation_mode	0b - Latency mode. 1b - Period mode, for bandwidth regulation.
19-17 —	Reserved
16 AWQOS_regulation_mode	AWQOS_regulation_mode Configures the mode of the QoS value regulator for write transactions: 0b - Latency mode. 1b - Period mode, for bandwidth regulation.
15-4 —	Reserved
3 AR_OT_regulation	AR_OT_regulation Enable regulation of outstanding read transactions for slave interfaces <ul style="list-style-type: none"> <li>• ACE-Lite interfaces only, for example S0, S1, and S2.</li> <li>• RAZ/WI for ACE interfaces, for example S3 and S4.</li> </ul>
2 AW_OT_regulation	AW_OT_regulation Enable regulation of outstanding write transactions for slave interfaces <ul style="list-style-type: none"> <li>• ACE-Lite interfaces only, for example S0, S1, and S2.</li> <li>• RAZ/WI for ACE interfaces, for example S3 and S4.</li> </ul>
1 ARQOS_regulation_read	ARQOS regulation read Enable QoS value regulation on reads for slave interfaces
0 AWQOS_regulation_write	AWQOS regulation write Enable QoS value regulation on writes for slave interfaces

## 10.2.12 Max OT Registers (Max\_OT\_Register\_S0 - Max\_OT\_Register\_S4)

### 10.2.12.1 Offset

Register	Offset
Max_OT_Register_S0	1110h
Max_OT_Register_S1	2110h
Max_OT_Register_S2	3110h
Max_OT_Register_S3	4110h
Max_OT_Register_S4	5110h

## 10.2.12.2 Function

The Max OT registers determine how many outstanding transactions are permitted when the OT regulator is enabled for each ACE-Lite slave interface. One register exists for each of the S0, S1, and S2 slave interfaces. A value of 0 for both the integer and fractional parts disables the programmable regulation so that the hardware limits apply. A value of 0 for the fractional part disables the regulation of fractional outstanding transactions. If int is the value of the integer part and frac is the value of the fractional part, then: Maximum mean number of outstanding transactions = int + frac/256.

Setting the maximum outstanding transaction size greater than that configured in the RTL, using the R\_MAX or W\_MAX parameters, has no effect. Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments.

Available in all CCI-400 configurations.

## 10.2.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				Int_OT_AR				Frac_OT_AR							
W																
Reset	u	u	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Int_OT_AW				Frac_OT_AW							
W																
Reset	u	u	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 10.2.12.4 Fields

Field	Function
31-30 —	Reserved
29-24	Int_OT_AR

Table continues on the next page...

Field	Function
Int_OT_AR	Integer part of the maximum outstanding AR addresses S0
23-16	Frac_OT_AR
Frac_OT_AR	Fractional part of the maximum outstanding AR addresses S0
15-14	Reserved
—	
13-8	Int_OT_AW
Int_OT_AW	Integer part of the maximum outstanding AW addresses S0
7-0	Frac_OT_AW
Frac_OT_AW	Fractional part of the maximum outstanding AW addresses S0

## 10.2.13 Regulator Target Registers (Target\_Latency\_Register\_S0 - Target\_Latency\_Register\_S4)

### 10.2.13.1 Offset

Register	Offset
Target_Latency_Register_S0	1130h
Target_Latency_Register_S1	2130h
Target_Latency_Register_S2	3130h
Target_Latency_Register_S3	4130h
Target_Latency_Register_S4	5130h

### 10.2.13.2 Function

The Regulator Target registers determines the target, in cycles, for the regulation of reads and writes. The target is either transaction latency or inter-transaction period, depending on the programming of the QoS Control Register. A value of 0 corresponds to no regulation. One register exists for each slave interface.

Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments.

Only has an effect when QOSOVERRIDE is HIGH for the associated interface.

### 10.2.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				AR_Lat											
W																
Reset	u	u	u	u	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				AW_Lat											
W																
Reset	u	u	u	u	0	0	0	0	0	0	0	0	0	0	0	0

### 10.2.13.4 Fields

Field	Function
31-28	Reserved
—	
27-16	AR channel target latency
AR_Lat	
15-12	Reserved
—	
11-0	AW channel target latency
AW_Lat	

## 10.2.14 QoS Range Register (Qos\_Range\_Register\_S0 - Qos\_Range\_Register\_S4)

### 10.2.14.1 Offset

Register	Offset
Qos_Range_Register_S0	1138h
Qos_Range_Register_S1	2138h
Qos_Range_Register_S2	3138h
Qos_Range_Register_S3	4138h
Qos_Range_Register_S4	5138h

### 10.2.14.2 Function

The QoS Range register enables you to program the minimum and maximum values for the ARQOS and AWQOS signals that the QV regulators generate. One register exists for each slave interface.

Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments.

Only has an effect when QOSOVERRIDE is HIGH for the associated interface.

### 10.2.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				Max_ARQOS				Reserved				Min_ARQOS			
Reset	u	u	u	u	0	0	0	0	u	u	u	u	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				Max_AWQOS				Reserved				Min_AWQOS			
Reset	u	u	u	u	0	0	0	0	u	u	u	u	0	0	0	0

### 10.2.14.4 Fields

Field	Function
31-28 —	Reserved
27-24 Max_ARQOS	Maximum ARQOS value
23-20 —	Reserved
19-16 Min_ARQOS	Minimum ARQOS value
15-12 —	Reserved
11-8	Maximum AWQOS value

Table continues on the next page...

Field	Function
Max_AWQOS	
7-4	Reserved
—	
3-0	Minimum AWQOS value
Min_AWQOS	

## 10.2.15 QoS Regulator Scale Factor Registers (Latency\_Regulation\_Register\_S0 - Latency\_Regulation\_Register\_S4)

### 10.2.15.1 Offset

Register	Offset
Latency_Regulation_Register_S0	2268h
Latency_Regulation_Register_S1	3268h
Latency_Regulation_Register_S2	4268h
Latency_Regulation_Register_S3	5268h
Latency_Regulation_Register_S4	6268h

### 10.2.15.2 Function

The QoS Regulator Scale Factor Registers characteristics are: Purpose QoS regulation value, AWQOS or ARQOS, scale factor coded for powers of 2 in the range  $2^{-5}$ - $2^{12}$ , to match a 16-bit integrator. One register exists for each slave interface.

Accessible using only Secure accesses, unless you set the Secure Access Register. See Secure Access Register bit assignments.

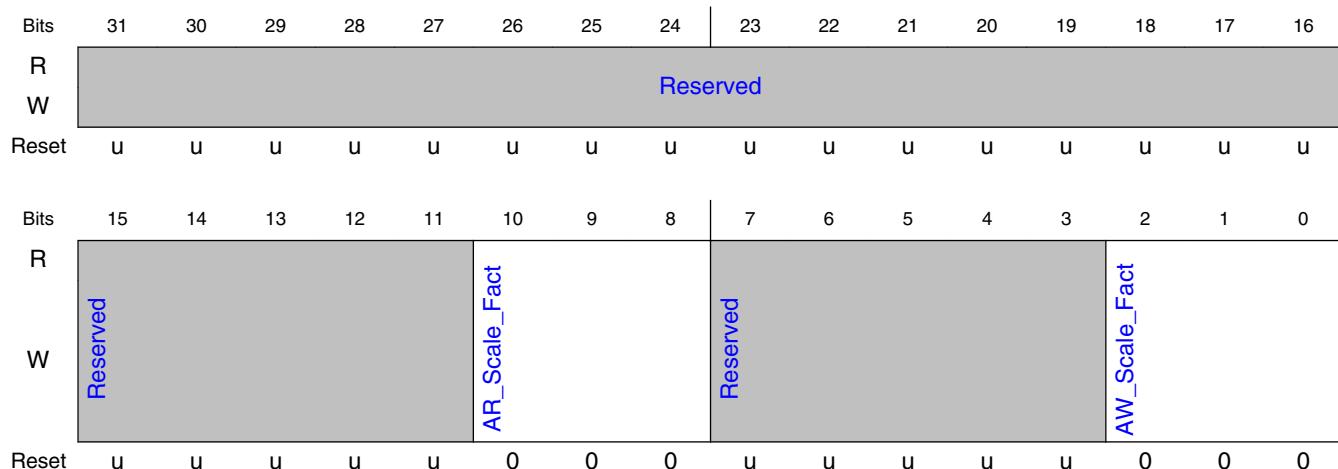
Only has an effect when QOSOVERRIDE is HIGH for the associated interface.

The table here shows the mapping of Scale Factor Register value to the scale factor.

**Table 10-2. Mapping of Scale Factor Register value to Regulator scale factor**

Scale Factor Register value	Scale factor
0x0	$2^{-5}$
0x1	$2^{-6}$
0x2	$2^{-7}$
0x3	$2^{-8}$
0x4	$2^{-9}$
0x5	$2^{-10}$
0x6	$2^{-11}$
0x7	$2^{-12}$

### 10.2.15.3 Diagram



### 10.2.15.4 Fields

Field	Function
31-11 —	Reserved
10-8 AR_Scale_Fact	ARQOS Scale Factor ARQOS scale factor, power of 2 in the range $2^{-5}$ - $2^{-12}$ .
7-3 —	Reserved
2-0 AW_Scale_Fact	AWQOS Scale Factor AWQOS scale factor, power of 2 in the range $2^{-5}$ - $2^{-12}$ .

## 10.3 Functional Description

This chapter describes the functionality of the CoreLink CCI-400 Cache Coherent Interconnect.

### 10.3.1 About the functions

The CCI-400 combines interconnect and coherency functions into a single module.

### 10.3.2 Snoop connectivity and control

The CCI-400 has a fully-connected snoop topology, so if they are enabled:

- Each ACE slave interface snoops the other ACE slave interface.
- ACE-Lite slave interfaces snoop both ACE slave interfaces.
- DVM messages are broadcast through all enabled slave interfaces.

Snooping and DVM message broadcast are disabled at reset, so you must enable the appropriate masters for snooping and DVM messages using the Snoop Control Registers before shareable or DVM messages are sent to the CCI-400. See [Snoop Control Registers \(Snoop\\_Control\\_Register\\_S0 - Snoop\\_Control\\_Register\\_S4\)](#).

Each ACE slave interface has programmable bits in the Snoop Control Registers. These bits control the issuing of snoop and DVM message requests on that interface.

#### NOTE

ACE-Lite slave interfaces have programmable bits to enable DVM messages only.

These programmable bits of the Snoop Control Registers are tied LOW at reset so you must program them HIGH for each master in the shareable domain before shareable transactions or DVM messages are sent to the CCI-400. Before disabling a master, you must disable snoop and DVM messages to that master by programming the relevant Snoop Control Register bits LOW.

To avoid deadlock through having AC requests enabled to interfaces where masters are not present, or not able to process them, the CCI-400 has the following hardware and software override mechanisms:

- Each slave interface has an **ACCHANELEN** input bit that, if you tie it LOW, prevents that interface from issuing any AC requests.

### NOTE

These bits are only sampled at reset.

- There are bits in the Control Override Register to disable all snooping or all DVM message broadcast, irrespective of the programming of the Snoop Control Register.

#### 10.3.2.1 Removing a master from the coherent domain

If you want to remove a master from the coherent domain, for example if a processor is being powered down, take the following actions:

1. Stop the processor from issuing shareable transactions. See the documentation of the processor.
2. Clean any shareable data in the processor cache. See the documentation of the processor.
3. Use the Snoop Control Register to prevent any more snoops or DVM messages being sent to the processor. See the [Snoop Control Registers \(Snoop\\_Control\\_Register\\_S0 - Snoop\\_Control\\_Register\\_S4\)](#).
4. Poll the CCI Status Register to confirm that the changes to the Snoop Control Register have completed. See the [Status Register \(Status\\_Register\)](#).

After you complete these actions, the master is no longer in the coherent domain and you can power it down or disable it. You must enable snoops to that master again before it allocates any shareable data in its cache.

#### 10.3.3 Speculative fetch

For an application where the probability of a miss is high, then the snoop request and response time adds directly to the latency for each transaction labelled as shareable. To mitigate this, you can program each master interface to issue a fetch downstream in parallel with issuing a snoop request. This is known as a *speculative fetch*.

In the event that the snoop associated with a speculative fetch hits in a cache, then the data from the snoop is returned in preference to the data from the speculative fetch.

A speculative fetch is issued before all hazards that had arisen from the corresponding snoop have been resolved. Therefore, it is sometimes necessary to discard the data returned from memory and retry the fetch. These cases are:

- When a hazard write transaction is detected. This hazard write transaction must be ordered before the speculative fetch.
- When data from the speculative fetch returns before the snoop response for that transaction, and the read data buffer is already occupied by data waiting for a snoop response.

You can use the PMU to record the number of retry transactions for each master interface.

### **NOTE**

Speculative fetches are only issued for these read-type transactions:

- ReadOnce .
- ReadClean .
- ReadNotSharedDirty .
- ReadUnique .
- ReadShared.

Although speculative fetches reduce the latency in the case of a snoop miss, there is a bandwidth and power penalty because of the additional transactions on a snoop hit.

Therefore, you can disable speculative fetches where you expect a significant number of snoops to hit. You can use the Speculation Control Register to disable speculative fetches for a master or a slave interface. For example, you can disable speculative fetches for transactions from a master that is not latency sensitive. See [Speculation Control Register \(Speculation\\_Control\\_Register\)](#).

## **10.3.4 Security**

If you are building a system based on the Secure and Non-secure capabilities that Arm TrustZone® technology provides, then you must consider security issues. This section describes:

- [Internal programmers view](#).
- [Security of master interfaces](#).

### **10.3.4.1 Internal programmers view**

With the exception of the PMU registers, the programmers view defaults to Secure access only, as follows:

- Non-secure read requests to Secure registers receive a DECERR response, **RRESP[1:0] == 0b11**, and zeroed data.
- Non-secure write requests to Secure registers receive a DECERR response, **BRESP[1:0] == 0b11** and are *Write-Ignored* (WI).

### NOTE

Some accesses might receive a response before they reach the CCI-400 registers and so do not receive a DECERR response nor affect the register values. An example of this is a cache maintenance operation that incorrectly addresses the CCI-400 register space.

You can override these security settings in the Secure Access Register. At reset, you can only access this using Secure requests, but if you write to it, this enables Non-secure access to all registers in the CCI-400 except for the Control Override Register and Secure Access Register. See [Control Override Register \(Control\\_OVERRIDE\\_Register\)](#) and [Secure Access Register \(Secure\\_Access\\_Register\)](#).

#### 10.3.4.2 Security of master interfaces

Transactions from the CCI-400 master interfaces always retain the security setting of the originating transactions. This applies to:

- Non-shareable transactions.
- Snoop misses.
- Speculative fetches.
- CCI-400-generated writes.

#### 10.3.5 Error responses

The CCI-400 uses a mixture of precise and imprecise signaling of error responses, where:

- Precise errors are signalled back to the master on the R and B channels for the precise transaction that caused the error.
- Imprecise errors are not signalled on R and B channels but are instead signalled using the **nERRORIRQ** output pin (See Table "Interrupt Assignments" in the "Interrupt Assignments" chapter). You can identify the interface that received the error response by reading the Imprecise Error Register. See [Imprecise Error Register \(Imprecise\\_Error\\_Register\)](#).

### 10.3.5.1 Imprecise errors

[Table 10-3](#) shows the errors that are signalled as imprecise. All other sources of error are signalled precisely.

#### NOTE

An error is signalled either precisely or imprecisely, but never both.

**Table 10-3. Imprecise errors**

Transaction causing error	Channel receiving error	Imprecise error indicator from
A ReadX snoop that misses in the cache and fetches data from downstream	CR	Slave interface receiving the CR response
Distributed Virtual Memory message	CR	Slave interface receiving the CR response
Speculative fetch that returns an error, but the snoop returns data	R	Master interface receiving the R response
Speculative fetch that must be retried	R	Master interface receiving the R response
Write that the CCI-400 generated	B	Master interface receiving the B response
Snoop error generated by a WriteLineUnique or WriteUnique transaction	CR	Master interface receiving the CR response
Write error for WriteUnique transactions that have been split but not the last transaction in the split sequence	B	Slave interface that received the transaction that was split

The CCI-400 generates a precise DECERR response in the case of a security violation on a CCI-400 register access. See [Imprecise Error Register \(Imprecise\\_Error\\_Register\)](#) and [Security](#).

### 10.3.6 Barriers

The CCI-400 supports all types of AMBA 4 barrier transactions. Each slave interface broadcasts these to every master interface, ensuring that intermediate transaction source and sink points observe the barrier correctly.

### 10.3.7 DVM messages

The ACE slave interfaces support DVM messages through their regular AC and CR channels. The ACE-Lite interfaces all contain AC and CR channels to support DVM messages only. Each slave interface has a programmable enable bit to determine whether it supports the issuing of AC requests for DVM messages. DVM messages are handled as regular transactions in the CCI-400, except that they are decoded based on the DVM message indicator, instead of the address, to ensure that multi-transaction DVM messages are correctly ordered.

The Snoop Control Registers and Control Override Register control the issuing of DVM message requests. See [Snoop Control Registers \(Snoop\\_Control\\_Register\\_S0 - Snoop\\_Control\\_Register\\_S4\)](#) and [Control Override Register \(Control\\_Override\\_Register\)](#).

ACE and ACE-Lite, plus DVM slave interfaces support all types of DVM transaction. These are:

- DVM Operation.
- DVM Synchronization.
- DVM Complete.

The R channel response to a DVM messages is sent immediately by the CCI-400. If the DVM message results in an error response, this is signaled imprecisely. For more information, see [Error responses](#) .

#### **NOTE**

- A master that issues DVM messages must also be able to receive DVM messages. The slave interface through which the master connects must have DVM messages enabled.

### **10.3.8 Quality of Service**

The CCI-400 supports QoS with the following independent mechanisms:

- [QoS value](#)
- [Regulation based on outstanding transactions](#) .

#### **10.3.8.1 QoS value**

Each CCI-400 slave interface has **ARQOS** and **AWQOS** input signals that transport a transaction-based QoS value. This determines the relative priority between transactions on that interface, or between interfaces. The CCI-400 uses the QoS value when it chooses

between transaction requests at arbitration points and within queues. Transaction requests with the highest QoS value are prioritized. The CCI-400 uses a *Least Recently Granted* (LRG) scheme when two or more transactions share the highest value.

QoS values are propagated by CCI-400.

### NOTE

Ensure that you balance the relative priorities of all slave interfaces. For example, setting each to the highest QoS value reduces the arbitration to LRG and no advantage is gained from having a QoS value.

You can override the **ARQOS** and **AWQOS** input signals from each slave interface by using a programmable register if the relevant static input signal, **QOSOVERRIDE[4:0]**, with one bit for each of slave interfaces 4-0, is HIGH. The QoS override is either based on a programmable value or uses performance feedback to set the value within a programmable range. Transactions that the CCI-400 generates use the same QoS value as the instigating transaction or the override value if **QOSOVERRIDE** is set.

### NOTE

**QOSOVERRIDE[4:0]** input signal is not controllable in this device and set to zero.

### NOTE

**QOSOVERRIDE** only applies to transactions that have a **ARQOS** or **AWQOS** value of 0. Therefore, each interface can have a mixture of traffic that is overridden or regulated and other traffic, with non-zero QoS value, that is unaffected. For example, high priority MMU page table requests might be mixed with high-bandwidth media requests that require regulation.

## QoS value regulation

CCI-400 regulation mechanisms vary the transaction QoS value depending on latency or bandwidth achieved through that slave interface. You can program target latency or bandwidth and a QoS value range for each regulator. Arm recommends that achievable targets are set so that the regulator uses the minimum QoS value in most cases and only increases the QoS value, up to the programmed maximum, under worst case conditions. The maximum value for each regulator is 0 at reset, so you must program a maximum value before the regulator can be used.

You can control the rate of change of the regulator integrator by using a programmable scale factor. There are two types of QoS value regulation:

- Regulation based on latency.
- Regulation based on bandwidth.

## Regulation based on latency

In this regulation mode, QoS values change based on measured latency. The value tends to increase if the latency is greater than the target and decrease if the latency is lower than the target.

## Regulation based on bandwidth

For bandwidth regulation, the target used for feedback is the period between successive request handshakes, in cycles. The value tends to increase if the period is greater than the target and decrease if the period is lower than the target. If the average number of bytes per request is known, this is equivalent to a bandwidth measure. Shareable transactions in the CCI-400 are 64 bytes in size, so this is usually a good approximation to use.

There are two modes of operation available when you are using this type of regulation:

### Normal mode

In this mode the QoS value remains stable when the master is idle, this is equivalent to measuring the average bandwidth only when the master is active. This is the default mode.

### Quiesce High mode

In this mode, the QoS value tends to the maximum programmed value when the master is idle, so when it becomes active, the initial transactions have a high QoS value. This mode is suitable for latency sensitive masters because it allows the master to be serviced with high priority while the bandwidth requirement is below that set. If the master starts to exceed its programmed bandwidth, the priority is reduced. You can use this mechanism to ensure that other masters are not excluded when latency sensitive masters take significant bandwidth.

You enable QoS value regulation by setting the appropriate control bits. See [Qos Control Register \(Qos\\_Control\\_Register\\_S0 - Qos\\_Control\\_Register\\_S4\)](#). When you enable QoS value regulation, ARQOS and AWQOS values are driven by those generated by the regulators, if the original transaction has a zero QoS value and the **QOSOVERRIDE** configuration input is HIGH.

You can program the regulator mode using the QoS Control Registers.

**NOTE**

- Turning QoS value regulation on when **QOSOVERRIDE[x]** is set LOW for a specific interface has no effect.
- Transactions that do not transfer data are not counted for QoS value regulation and do not have their QoS value overridden. These transactions are:
  - CleanUnique .
  - MakeUnique .
  - CleanShared .
  - CleanInvalid .
  - MakeInvalid .
  - Evict .
  - Barriers.
  - DVM transactions.

### **10.3.8.2 Regulation based on outstanding transactions**

The CCI-400 offers an additional mechanism for regulating traffic flows for the benefit of overall performance. Each ACE-Lite slave interface has an optional, programmable mechanism for limiting the number of outstanding read and write transactions, where an *Outstanding Transaction* (OT) is a read request without read data, or a write request without a response. This can be used where QoS value regulation is not effective because the system is not sensitive to QoS value. The disadvantage of this form of regulation is that it might stall the master even when the system is idle and traffic from this master is not affecting the performance of other masters.

You can characterize a sequence of transactions, with periods when there are no outstanding transactions, by using a fractional outstanding transaction number. For example, if requests occur every 100 cycles, but it only takes 50 cycles for the last response to arrive, then this corresponds to 0.5 OTs. The sum of the integer and fractional values represents a maximum of the mean number of OTs in a sliding window and, consequently, also over all time. If the fractional part is 0, the number of OTs is never permitted to exceed the integer part. If the fractional part is not 0, the number of OTs is not permitted to exceed one more than the integer part. This mean value is only achieved if the attached master maintains the maximum number of transactions it is able to issue at all times. Therefore, if the integer part is 0 and the fractional part is 0.5, and transactions have a lifespan of 50 cycles, then a master can issue a transaction. It finishes after 50 cycles and it cannot issue the next transaction until 100 cycles, maintaining a mean number of outstanding transactions as 0.5.

If you enable regulation, the following programmable values set the permitted number of outstanding transactions:

- OT integer.
- OT fraction.

### **NOTE**

- Outstanding transaction regulation only counts transactions with a zero QoS value.
- Outstanding transaction regulation does not count or override transactions that have no data associated with them. These transactions are:
  - CleanUnique .
  - MakeUnique .
  - CleanShared .
  - CleanInvalid .
  - MakeInvalid .
  - Evict .
  - Barriers.
  - DVM transactions.

### **Read Queue Slot Reservation**

Each master interface of the CCI-400 has a queue that stores read requests. If this becomes full with low priority requests, higher priority requests are blocked. To avoid this, the CCI-400 reserves a number of slots for high priority requests and a number of slots for high or medium priority requests. Table [Regulation based on outstanding transactions](#) shows a summary of the possible configurations.

**Table 10-4. Slots reserved for high and medium priority traffic in each master interface**

Read tracker size, as determined by the <code>MIX_R_MAX</code> parameter	Number of reserved slots for medium or high priority requests	Number of reserved slots for only high priority requests
2-4	0	0
5-7	0	1
8-15	1	1
16-128	3	1

For example, assume the read tracker size is 32, implying that three slots are reserved for medium to high priority requests and one slot is reserved for high priority requests. In this case:

- The maximum number of slots available for high priority requests is 32.
- The maximum number of slots available for medium priority requests is  $32 - 1 = 31$ .
- The maximum number of slots available for low priority requests is  $32 - 3 - 1 = 28$ .

The QoS values that are considered as high and medium priority can be configured at the design time using the R\_THRESHOLD\_UPPER and R\_THRESHOLD\_LOWER parameters however the values configured for these two parameters in this chip are 15 and 11.

### **10.3.8.3 QoS programmable registers**

Programmers Model describes the following registers:

- Read Channel QoS Value Override Register.
- Write Channel QoS Value Override Register.
- QoS Control Register.
- Max OT Registers.
- Regulator Target Registers.
- QoS Regulator Scale Factor Registers.

# Chapter 11

## Arm CoreLink™ TrustZone Address Space Controller TZC-380

### 11.1 Introduction

This chapter introduces the CoreLink TrustZone Address Space Controller (TZC-380).

#### 11.1.1 Overview

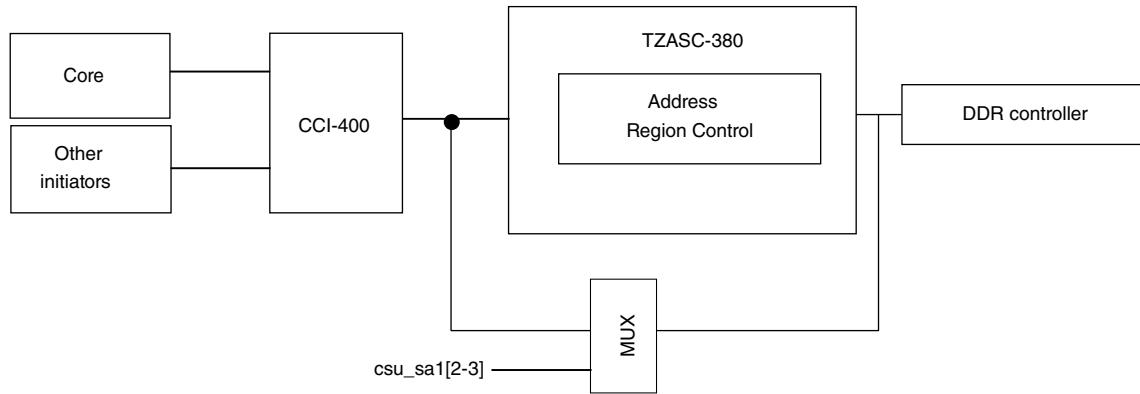
The TZASC is an advanced microcontroller bus architecture (AMBA) compliant system-on-chip (SoC) peripheral. It is a high-performance, area-optimized address space controller with on-chip AMBA bus interfaces that conform to the AMBA advanced eXtensible interface (AXI) protocol and the AMBA advanced peripheral bus (APB) protocol.

The TZASC can be configured to provide the optimum security address region control functions required for intended application. See [Features](#) for a summary of the configurable features supported.

#### NOTE

The software should configure the TZASC bypass mux (csu\_sa1[2-3]) to disable the bypass operation and use the TZASC default region before any transaction goes to DDR.

The figure below shows the TZASC in an example system:

**Figure 11-1. Example system**

### 11.1.1.1 Features

The TZASC provides the following features:

- Programs security access permissions for each address region
- Transfers data between master and slave only if the security status of the AXI transaction matches the security settings of the memory region it addresses
- Prevents write access to various registers after assertion of `secure_boot_lock`, which is controlled by the CSU\_SA1[4:5] bit. Refer [CSU Memory Map/Register Definition](#) for details.

The number of address regions can be configured to:

- 2 regions
- 4 regions
- 8 regions
- 16 regions

## 11.2 Miscellaneous signal descriptions

There are two miscellaneous signals provided by TZASC:

- **`secure_boot_lock`**. The assertion of this signal is controlled by the CSU\_SA1[4:5] bit. Refer [CSU Memory Map/Register Definition](#) for details.
- **`tzasc_int`**. The assertion of this signal is controlled by the TZASC interrupt (#125). Refer [Internal interrupt sources](#) for details.

The figure below shows the signals provided by the TZASC:



**Figure 11-2. Miscellaneous signals**

Asserting `secure_boot_lock` enhances the security of the TZASC. See [Preventing writes to registers and using `secure\_boot\_lock`](#).

You can program the TZASC to assert `tzasc_int` when it denies an AXI master access to a region. See [Denied AXI transactions](#).

## 11.3 Register Descriptions

The following information applies to the TZASC registers:

- The base address of the TZASC is not fixed, and can be different for any particular system implementation. The offset of each register from the base address is fixed.
- Do not attempt to access reserved or unused address locations. Attempting to access these location can result in Unpredictable behavior of the TZASC.
- Unless otherwise stated in the accompanying text:
  - do not modify undefined register bits
  - ignore undefined register bits on reads
  - all register bits are reset to a logic 0 by a system or power-on reset.
- For programming the registers, the TZASC supports data in word-invariant endianness.
- System designers must ensure that only processors in Secure state can access the registers, otherwise it can compromise the security of the system.

### NOTE

See [Constraints of use](#) for more information about considerations relating to change in programmers view on an active system.

The TZASC register map consists of the following regions:

- Configuration, lockdown, and interrupt : Use these registers to determine the global configuration of the TZASC, and control its operating state.
- Fail status : These registers provide information about an access that failed because of insufficient permissions.
- Control : Use these registers to enable the TZASC to perform security inversion or speculative accesses.

## Register Descriptions

- Region control : Use these registers to control the operating state of each region.
- Integration test : Use these registers when testing the integration of the TZASC in a System-on-Chip (SoC).
- Component configuration : These registers enable the identification of system components by software.

### 11.3.1 TZASC memory map

TZASC base address: 150\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration Register (configuration)	32	RO	0000_270Fh
4h	Action register (action)	32	RW	0000_0001h
8h	Lockdown Range Register (lockdown_range)	32	RW	0000_0000h
Ch	Lockdown Select Register (lockdown_select)	32	RW	0000_0000h
10h	Interrupt Status Register (int_status)	32	RO	0000_0000h
14h	Interrupt Clear Register (int_clear)	32	WO	0000_0000h
20h	Fail Address Low Register (fail_address_low)	32	RO	0000_0000h
24h	Fail Address High Register (fail_address_high)	32	RO	0000_0000h
28h	Fail Control Register (fail_control)	32	RO	0000_0000h
2Ch	Fail ID Register (fail_id)	32	RO	0000_0000h
30h	Speculation Control Register (speculation_control)	32	RW	0000_0000h
34h	Security Inversion Register (security_inversion_en)	32	RW	0000_0000h
100h	Region Setup Low 0 Register (region_setup_low_0)	32	RO	0000_0000h
104h	Region Setup High 0 Register (region_setup_high_0)	32	RO	0000_0000h
108h	Region Attributes 0 Register (region_attributes_0)	32	RW	C000_0000h
110h	Region Setup Low 1 Register (region_setup_low_1)	32	RO	0000_0000h
114h	Region Setup High 1 Register (region_setup_high_1)	32	RO	0000_0000h
118h	Region Attributes 1 Register (region_attributes_1)	32	RW	0000_001Ch
120h	Region Setup Low 2 Register (region_setup_low_2)	32	RO	0000_0000h
124h	Region Setup High 2 Register (region_setup_high_2)	32	RO	0000_0000h
128h	Region Attributes 2 Register (region_attributes_2)	32	RW	0000_001Ch
130h	Region Setup Low 3 Register (region_setup_low_3)	32	RO	0000_0000h
134h	Region Setup High 3 Register (region_setup_high_3)	32	RO	0000_0000h
138h	Region Attributes 3 Register (region_attributes_3)	32	RW	0000_001Ch
140h	Region Setup Low 4 Register (region_setup_low_4)	32	RO	0000_0000h
144h	Region Setup High 4 Register (region_setup_high_4)	32	RO	0000_0000h
148h	Region Attributes 4 Register (region_attributes_4)	32	RW	0000_001Ch
150h	Region Setup Low 5 Register (region_setup_low_5)	32	RO	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
154h	Region Setup High 5 Register (region_setup_high_5)	32	RO	0000_0000h
158h	Region Attributes 5 Register (region_attributes_5)	32	RW	0000_001Ch
160h	Region Setup Low 6 Register (region_setup_low_6)	32	RO	0000_0000h
164h	Region Setup High 6 Register (region_setup_high_6)	32	RO	0000_0000h
168h	Region Attributes 6 Register (region_attributes_6)	32	RW	0000_001Ch
170h	Region Setup Low 7 Register (region_setup_low_7)	32	RO	0000_0000h
174h	Region Setup High 7 Register (region_setup_high_7)	32	RO	0000_0000h
178h	Region Attributes 7 Register (region_attributes_7)	32	RW	0000_001Ch
180h	Region Setup Low 8 Register (region_setup_low_8)	32	RO	0000_0000h
184h	Region Setup High 8 Register (region_setup_high_8)	32	RO	0000_0000h
188h	Region Attributes 8 Register (region_attributes_8)	32	RW	0000_001Ch
190h	Region Setup Low 9 Register (region_setup_low_9)	32	RO	0000_0000h
194h	Region Setup High 9 Register (region_setup_high_9)	32	RO	0000_0000h
198h	Region Attributes 9 Register (region_attributes_9)	32	RW	0000_001Ch
1A0h	Region Setup Low 10 Register (region_setup_low_10)	32	RO	0000_0000h
1A4h	Region Setup High 10 Register (region_setup_high_10)	32	RO	0000_0000h
1A8h	Region Attributes 10 Register (region_attributes_10)	32	RW	0000_001Ch
1B0h	Region Setup Low 11 Register (region_setup_low_11)	32	RO	0000_0000h
1B4h	Region Setup High 11 Register (region_setup_high_11)	32	RO	0000_0000h
1B8h	Region Attributes 11 Register (region_attributes_11)	32	RW	0000_001Ch
1C0h	Region Setup Low 12 Register (region_setup_low_12)	32	RO	0000_0000h
1C4h	Region Setup High 12 Register (region_setup_high_12)	32	RO	0000_0000h
1C8h	Region Attributes 12 Register (region_attributes_12)	32	RW	0000_001Ch
1D0h	Region Setup Low 13 Register (region_setup_low_13)	32	RO	0000_0000h
1D4h	Region Setup High 13 Register (region_setup_high_13)	32	RO	0000_0000h
1D8h	Region Attributes 13 Register (region_attributes_13)	32	RW	0000_001Ch
1E0h	Region Setup Low 14 Register (region_setup_low_14)	32	RO	0000_0000h
1E4h	Region Setup High 14 Register (region_setup_high_14)	32	RO	0000_0000h
1E8h	Region Attributes 14 Register (region_attributes_14)	32	RW	0000_001Ch
1F0h	Region Setup Low 15 Register (region_setup_low_15)	32	RO	0000_0000h
1F4h	Region Setup High 15 Register (region_setup_high_15)	32	RO	0000_0000h
1F8h	Region Attributes 15 Register (region_attributes_15)	32	RW	0000_001Ch
E00h	Integration Test Control Register (itcrg)	32	RW	0000_0000h
E04h	Integration Test Input Register (itip)	32	RO	0000_0000h
E08h	Integration Test Output Register (itop)	32	RW	0000_0000h

## 11.3.2 Configuration Register (configuration)

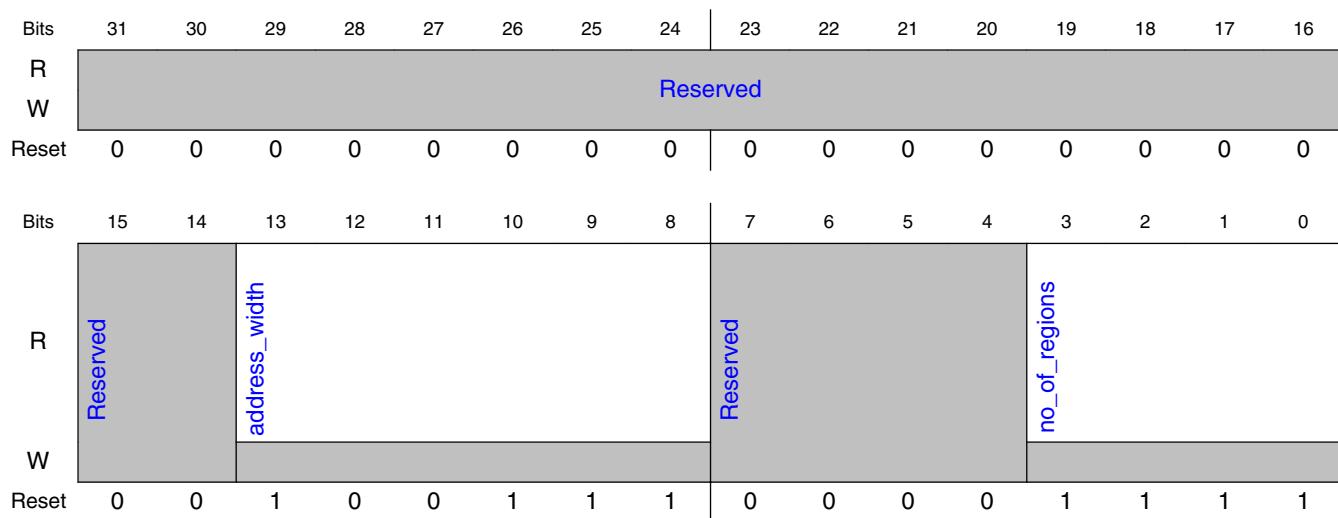
### 11.3.2.1 Offset

Register	Offset
configuration	0h

### 11.3.2.2 Function

The configuration Register provides information about the configuration of the TZASC. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.2.3 Diagram



### 11.3.2.4 Fields

Field	Function
31-14 —	Reserved, Should be Zero (SBZ).
13-8 address_width	Returns the width of the AXI address bus. Read as:

*Table continues on the next page...*

Field	Function
	b000000-b011110 = reserved b011111 = 32-bit b100000 = 33-bit b100001 = 34-bit ... b111110 = 63-bit b111111 = 64-bit.
7-4	Reserved, Should be Zero (SBZ).
—	
3-0 no_of_regions	no_of_regions Returns the number of regions that the TZASC provides: b0000 = reserved b0001 = 2 regions b0010 = 3 regions b0011 = 4 regions ... b1111 = 16 regions.

### 11.3.3 Action register (action)

#### 11.3.3.1 Offset

Register	Offset
action	4h

#### 11.3.3.2 Function

The action Register controls the response signaling behavior of the TZASC to region permission failures. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.3.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved															reaction_value	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	1

### 11.3.3.4 Fields

Field	Function
31-2	Reserved, Should be Zero (SBZ).
—	
1-0 reaction_value	<p>reaction_value</p> <p>Controls how the TZASC uses the bresps[1:0], rresps[1:0], and tzasc_int signals when a region permission failure occurs:</p> <ul style="list-style-type: none"> <li>b00 = sets tzasc_int LOW and issues an OKAY response</li> <li>b01 = sets tzasc_int LOW and issues a DECERR response</li> <li>b10 = sets tzasc_int HIGH and issues an OKAY response</li> <li>b11 = sets tzasc_int HIGH and issues a DECERR response.</li> </ul>

## 11.3.4 Lockdown Range Register (lockdown\_range)

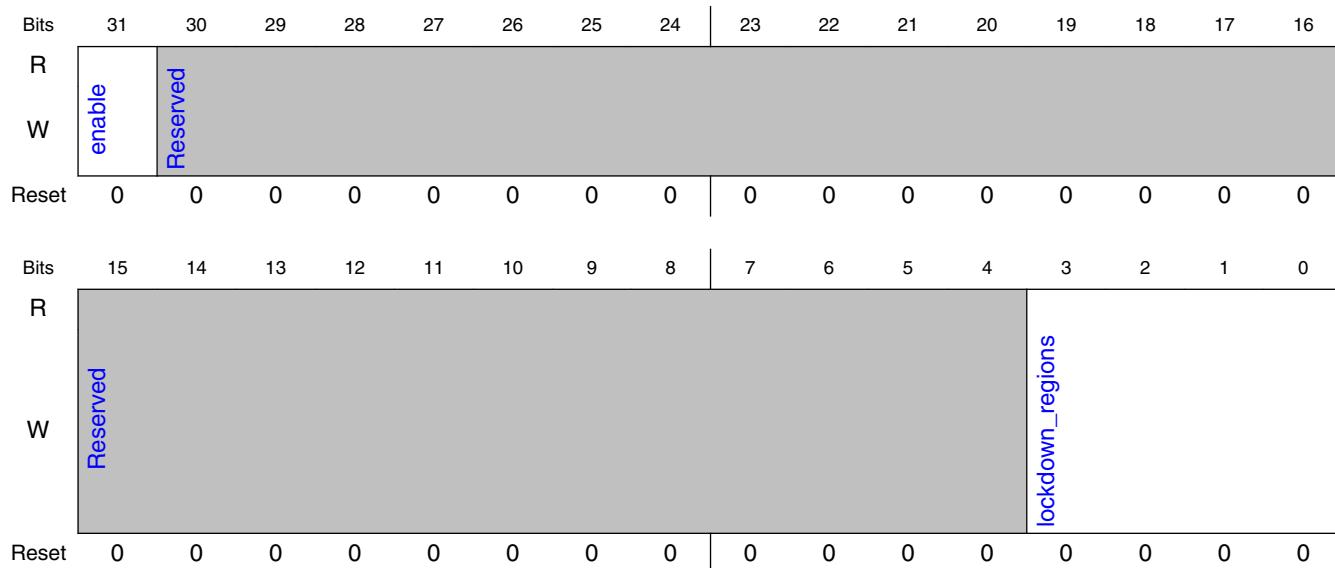
### 11.3.4.1 Offset

Register	Offset
lockdown_range	8h

### 11.3.4.2 Function

The lockdown\_range Register controls the range of regions that are locked down. The lockdown\_select Register can restrict the access type of this register to RO. Available in all configurations of the TZASC.

### 11.3.4.3 Diagram



### 11.3.4.4 Fields

Field	Function
31 enable	When set to 1, it enables the lockdown_regions field to control the regions that are to be locked.
30-4 —	Reserved, Should be Zero (SBZ).
3-0 lockdown_regions	<p>Controls the number of regions to lockdown when the enable bit is set to 1:</p> <p>b0000 = <i>region no_of_regions</i>-1 is locked</p> <p>b0001 = <i>region no_of_regions</i>-1 to <i>region no_of_regions</i>-2 are locked</p> <p>b0010 = <i>region no_of_regions</i>-1 to <i>region no_of_regions</i>-3 are locked</p> <p>b0011 = <i>region no_of_regions</i>-1 to <i>region no_of_regions</i>-4 are locked</p> <p>...</p> <p>b1111 = <i>region no_of_regions</i>-1 to <i>region no_of_regions</i>-16 are locked.</p>

## Register Descriptions

Field	Function
	<p><i>no_of_regions</i> is the value of the no_of_regions field in the configuration Register. See Configuration Register.</p> <p><b>NOTE:</b> The value programmed in lockdown_range Register must not be greater than no_of_regions-1 else all regions are locked.</p>

## 11.3.5 Lockdown Select Register (`lockdown_select`)

### 11.3.5.1 Offset

Register	Offset
<code>lockdown_select</code>	Ch

### 11.3.5.2 Function

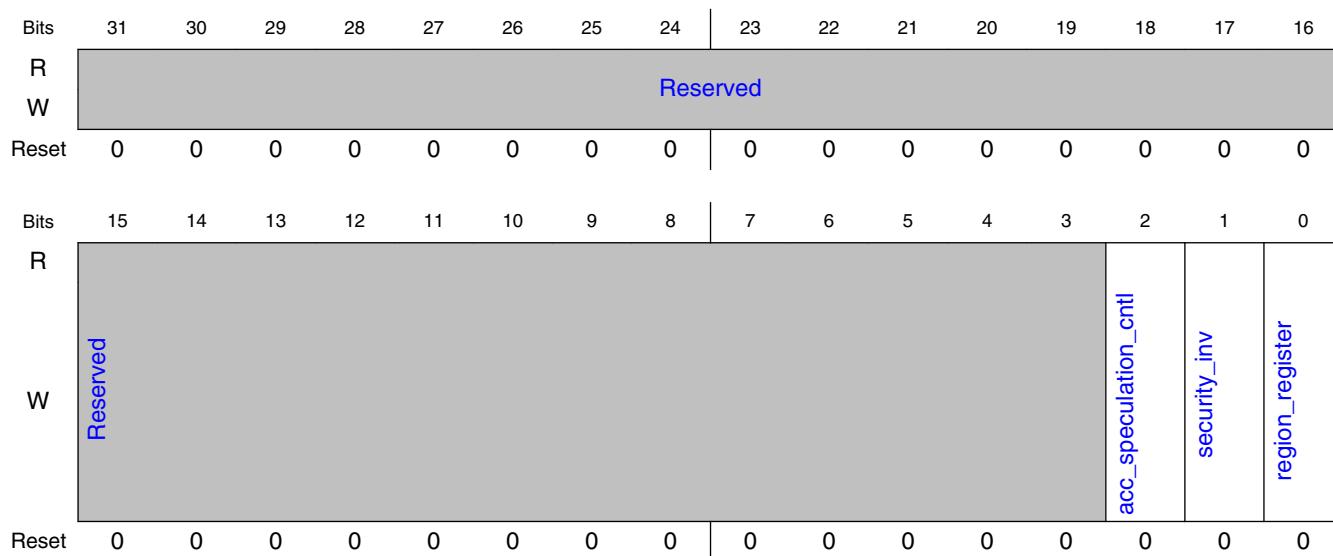
The `lockdown_select` Register controls whether the TZASC permits write accesses to the following registers:

- Lockdown Range Register
- Speculation Control Register
- Security Inversion Enable Register

After `aresetn` goes HIGH, the TZASC only permits write access to this register, if `secure_boot_lock` remains LOW. When `secure_boot_lock` is HIGH for one `aclk` period, or more then the TZASC ignores writes to this register.

This register is available in all configurations of the TZASC.

### 11.3.5.3 Diagram



### 11.3.5.4 Fields

Field	Function
31-3 —	Reserved, Should be Zero (SBZ).
2 acc_speculation_cntl	<p>acc_speculation_cntl</p> <p>Modifies the access type of the speculation_control Register:</p> <p>0 = no effect. speculation_control Register remains RW.</p> <p>1 = speculation_control Register is RO.</p> <p>See Speculation Control Register for more information.</p>
1 security_inv	<p>security_inv</p> <p>Modifies the access type of the security_inversion_en Register:</p> <p>0 = no effect. security_inversion_en Register remains RW.</p> <p>1 = security_inversion_en Register is RO.</p> <p>See Security Inversion Enable Register for more information.</p>
0 region_register	<p>region_register</p> <p>Modifies the access type of the lockdown_range Register:</p> <p>0 = no effect. lockdown_range Register remains RW.</p> <p>1 = lockdown_range Register is RO.</p> <p>See Lockdown Range Register for more information.</p>

## 11.3.6 Interrupt Status Register (int\_status)

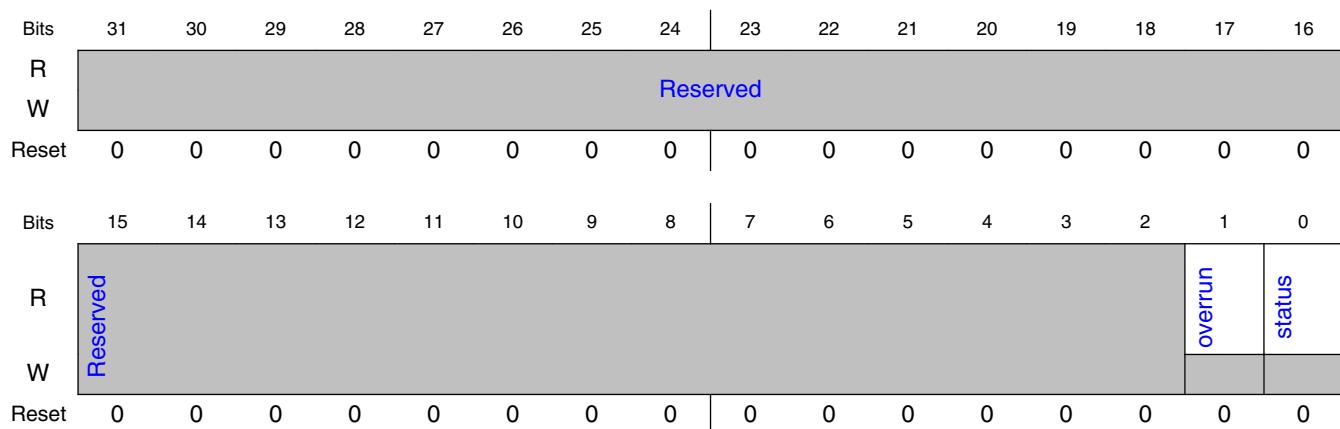
### 11.3.6.1 Offset

Register	Offset
int_status	10h

### 11.3.6.2 Function

The int\_status register characteristics are: Returns the status of the interrupt. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.6.3 Diagram



### 11.3.6.4 Fields

Field	Function
31-2	Reserved, Should be Zero (SBZ).
—	
1 overrun	overrun When set to 1, it indicates the occurrence of two or more region permission failures since the interrupt was last cleared

Table continues on the next page...

Field	Function
0 status	status Returns the status of the interrupt: 0 = interrupt is inactive 1 = interrupt is active.

## 11.3.7 Interrupt Clear Register (int\_clear)

### 11.3.7.1 Offset

Register	Offset
int_clear	14h

### 11.3.7.2 Function

The int\_clear Register clears the interrupt. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.7.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R																	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.7.4 Fields

Field	Function
31-0 int_clear	<p>int_clear</p> <p>Writing any value to the int_clear Register sets the:</p> <ul style="list-style-type: none"> <li>status bit to 0 in the int_status Register</li> <li>overrun bit to 0 in the int_status Register.</li> </ul> <p>See Interrupt Status register for details.</p>

## 11.3.8 Fail Address Low Register (fail\_address\_low)

### 11.3.8.1 Offset

Register	Offset
fail_address_low	20h

### 11.3.8.2 Function

The fail\_address\_low Register returns the address, the lower 32-bits, of the first access that failed a region permission, after the interrupt was cleared. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	add_status_low															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	add_status_low															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 11.3.8.4 Fields

Field	Function
31-0 add_status_low	Returns the AXI address bits [31:0] of the first access to fail a region permission check after the interrupt was cleared.

## 11.3.9 Fail Address High Register (fail\_address\_high)

### 11.3.9.1 Offset

Register	Offset
fail_address_high	24h

### 11.3.9.2 Function

The fail\_address\_high Register returns the address, the upper 32-bits, of the first access that failed a region permission, after the interrupt was cleared. There are no usage constraints. Only available when the TZASC has an AXI address width of greater than 32 bits.

### 11.3.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	add_status_high															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	add_status_high															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 11.3.9.4 Fields

Field	Function
31-0 add_status_high	add_status_high Returns the address bits [AXI_ADDRESS_MSB:32] of the first access to fail a region permission check after the interrupt was cleared. The size of this bitfield varies as [n:0] where n = AXI_ADDRESS_MSB – 32..

## 11.3.10 Fail Control Register (fail\_control)

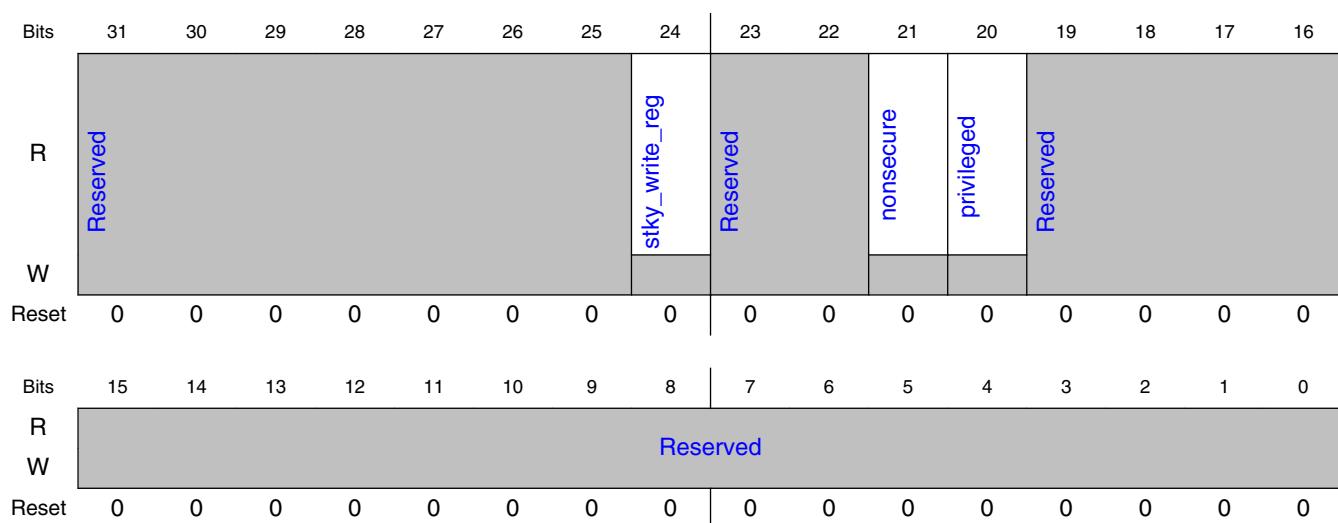
### 11.3.10.1 Offset

Register	Offset
fail_control	28h

### 11.3.10.2 Function

The fail\_control Register returns the control status information of the first access that failed a region permission, after the interrupt was cleared. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.10.3 Diagram



### 11.3.10.4 Fields

Field	Function
31-25 —	Reserved, Should be Zero (SBZ).
24 stky_write_reg	stky_write_reg This bit indicates whether the first access to fail a region permission check was a write or read as: 0 = read access 1 = write access.
23-22 —	Reserved, Should be Zero (SBZ).
21 nonsecure	nonsecure After clearing the interrupt status, this bit indicates whether the first access to fail a region permission check was non-secure. Read as: 0 = secure access 1 = non-secure access.
20 privileged	privileged After clearing the interrupt status, this bit indicates whether the first access to fail a region permission check was privileged. Read as: 0 = unprivileged access 1 = privileged access.
19-0 —	Reserved, Should be Zero (SBZ).

### 11.3.11 Fail ID Register (fail\_id)

#### 11.3.11.1 Offset

Register	Offset
fail_id	2Ch

### 11.3.11.2 Function

The fail\_id Register returns the master AXI ID of the first access that failed a region permission, after the interrupt was cleared. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.11.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved					fail_id											
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.11.4 Fields

Field	Function
31-12 —	Reserved, Should be Zero (SBZ).
11-0 fail_id	Returns the master AXI ID of the first access to fail a region permission check after the interrupt was cleared. The size of this bit field is [n:0] where n = AID_WIDTH-1.

## 11.3.12 Speculation Control Register (speculation\_control)

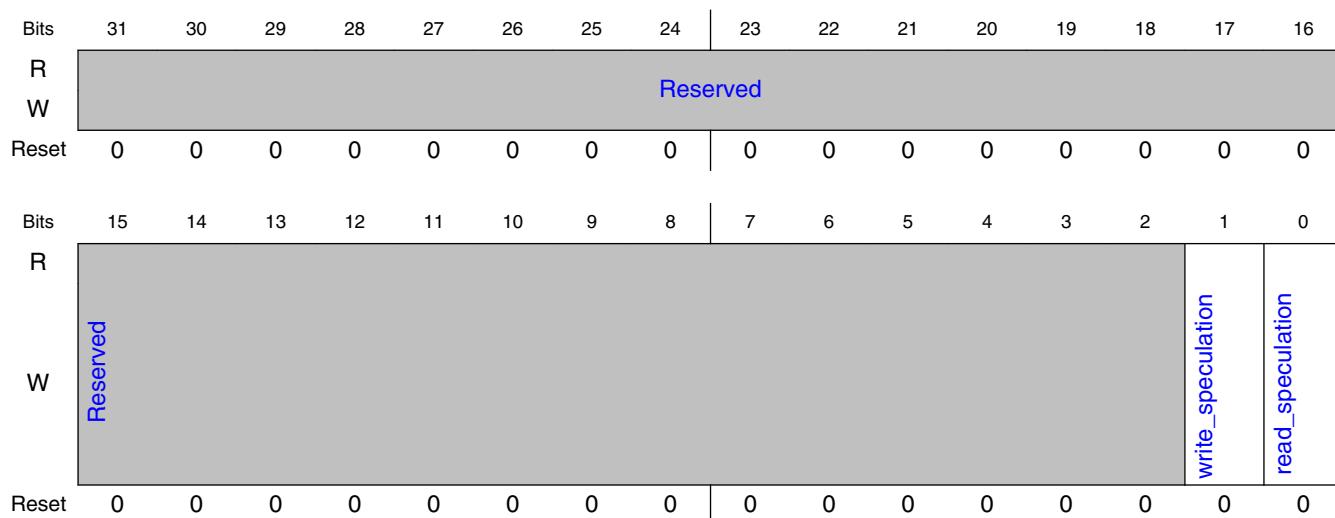
### 11.3.12.1 Offset

Register	Offset
speculation_control	30h

### 11.3.12.2 Function

The speculation\_control Register controls the read access speculation and write access speculation. The lockdown\_select Register can restrict the access type of this register to RO. See Lockdown Select Register for more details. Available in all configurations of the TZASC.

### 11.3.12.3 Diagram



### 11.3.12.4 Fields

Field	Function
31-2	Reserved, Should be Zero (SBZ).
—	
1 write_speculatio n	<p>write_speculation</p> <p>Controls the write access speculation:</p> <p>0 = write access speculation is enabled. This is the default.</p> <p>1 = write access speculation is disabled.</p>
0 read_speculatio n	<p>read_speculation</p> <p>Controls the read access speculation:</p> <p>0 = read access speculation is enabled. This is the default.</p> <p>1 = read access speculation is disabled.</p>

## 11.3.13 Security Inversion Register (`security_inversion_en`)

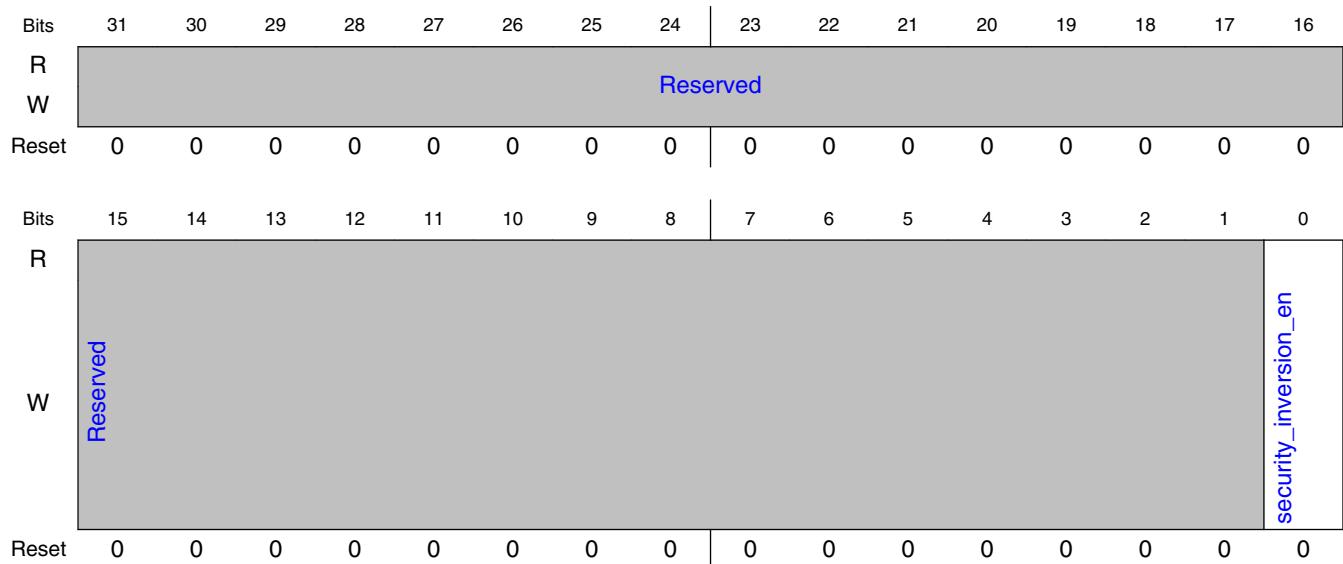
### 11.3.13.1 Offset

Register	Offset
<code>security_inversion_en</code>	34h

### 11.3.13.2 Function

The `security_inversion_en` Register controls whether the TZASC enables security inversion to occur. Usage constraints The lockdown\_select Register can restrict the access type of this register to RO. See Lockdown Select Register for more details. Available in all configurations of the TZASC.

### 11.3.13.3 Diagram



### 11.3.13.4 Fields

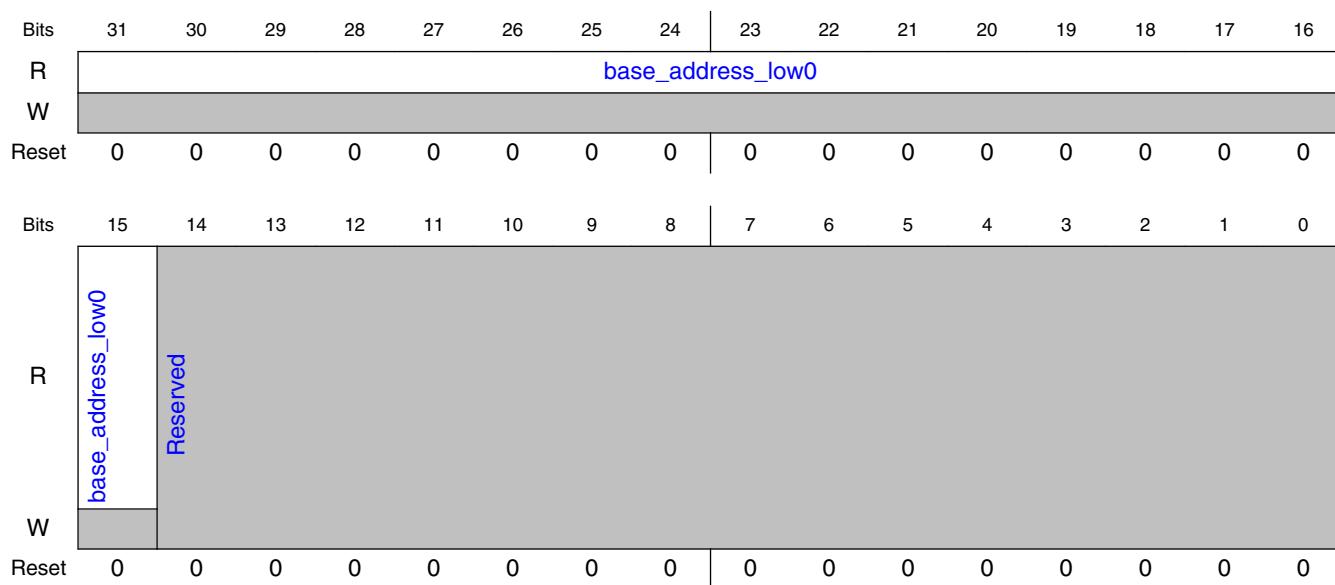
Field	Function
31-1 —	Reserved, Should be Zero (SBZ).
0 security_inversion_en	Controls whether the TZASC permits security inversion to occur: 0 = security inversion is not permitted. This is the default. 1 = security inversion is permitted. This enables a region to be accessible to masters in Non-secure state but not accessible to masters in Secure state.

### 11.3.14 Region Setup Low 0 Register (region\_setup\_low\_0)

#### 11.3.14.1 Offset

Register	Offset
region_setup_low_0	100h

#### 11.3.14.2 Diagram



### 11.3.14.3 Fields

Field	Function
31-15 base_address_low0	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.15 Region Setup High 0 Register (region\_setup\_high\_0)

### 11.3.15.1 Offset

Register	Offset
region_setup_high_0	104h

### 11.3.15.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.15.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high0																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high0																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.15.4 Fields

Field	Function
31-0 base_address_high0	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.16 Region Attributes 0 Register (region\_attributes\_0)

### 11.3.16.1 Offset

Register	Offset
region_attributes_0	108h

### 11.3.16.2 Function

The region\_attributes\_0 register controls the permissions for region 0. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.16.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16	
R	sp0									Reserved								
W																		
Reset	1	1	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0	
R	Reserved																	
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	

### 11.3.16.4 Fields

Field	Function
31-28	sp0
sp0	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the <i>spn</i> field controls whether the TZASC permits the transaction to proceed.
27-0 —	Reserved, Should be Zero (SBZ).

## 11.3.17 Region Setup Low 1 Register (region\_setup\_low\_1)

### 11.3.17.1 Offset

Register	Offset
region_setup_low_1	110h

### 11.3.17.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low1																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low1																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.17.3 Fields

Field	Function
31-15 base_address_low1	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.18 Region Setup High 1 Register (region\_setup\_high\_1)

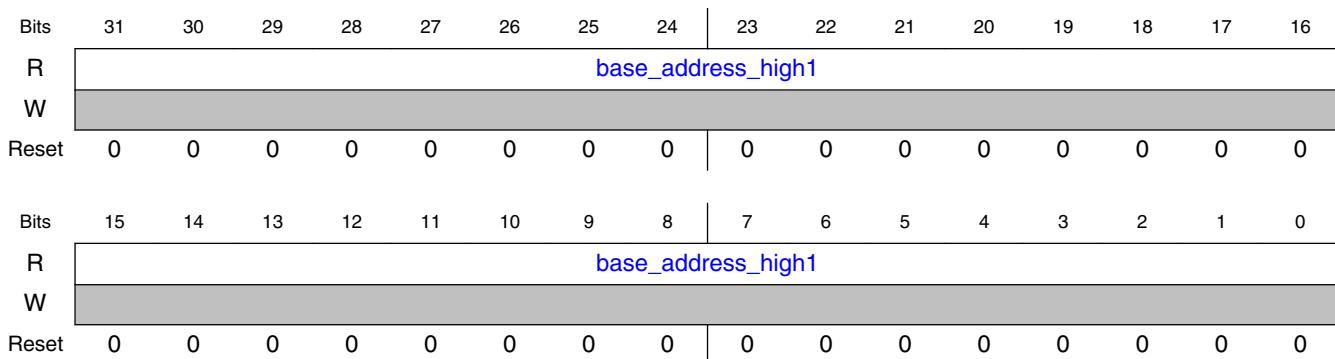
### 11.3.18.1 Offset

Register	Offset
region_setup_high_1	114h

### 11.3.18.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.18.3 Diagram



### 11.3.18.4 Fields

Field	Function
31-0 base_address_high1	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.19 Region Attributes 1 Register (region\_attributes\_1)

### 11.3.19.1 Offset

Register	Offset
region_attributes_1	118h

### 11.3.19.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-1. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-1. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.19.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16	
R	sp1									Reserved								
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0	
R	subregion_disable1									Reserved		size1				en1		
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	1	1	1	0	0	

### 11.3.19.4 Fields

Field	Function
31-28 sp1	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the <i>spn</i> field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable1	Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size1	size1

Table continues on the next page...

## Register Descriptions

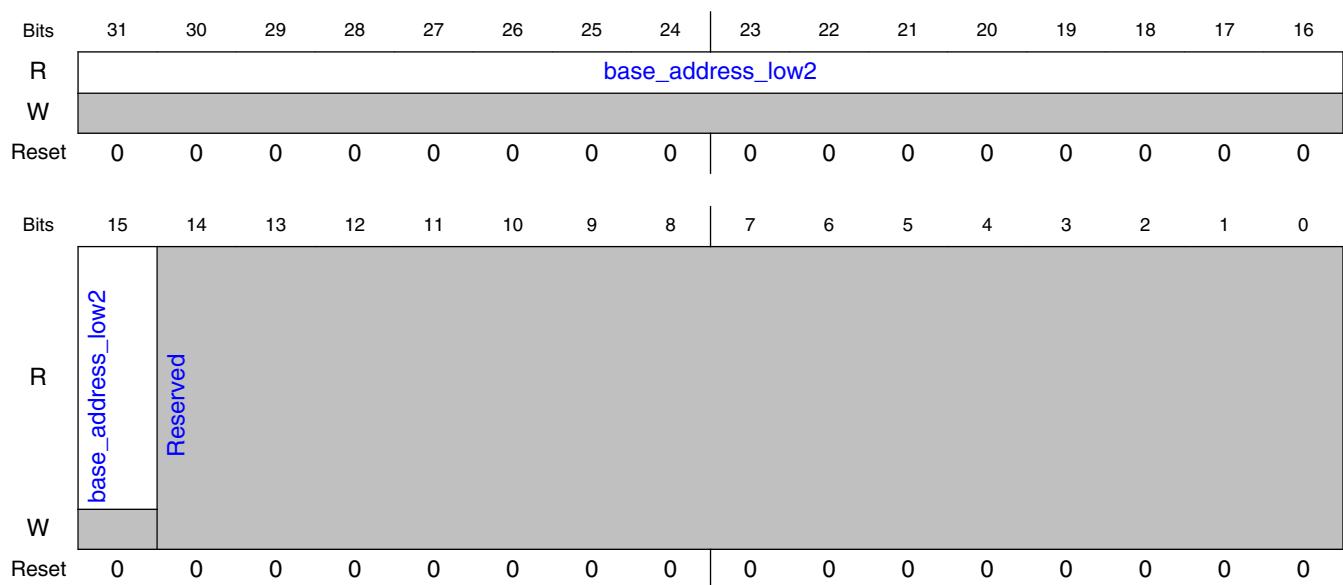
Field	Function
	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en1	<p>en1</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.20 Region Setup Low 2 Register (region\_setup\_low\_2)

### 11.3.20.1 Offset

Register	Offset
region_setup_low_2	120h

### 11.3.20.2 Diagram



### 11.3.20.3 Fields

Field	Function
31-15 base_address_low2	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.21 Region Setup High 2 Register (region\_setup\_high\_2)

#### 11.3.21.1 Offset

Register	Offset
region_setup_high_2	124h

#### 11.3.21.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.21.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high2																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high2																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.21.4 Fields

Field	Function
31-0 base_address_high2	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.22 Region Attributes 2 Register (region\_attributes\_2)

### 11.3.22.1 Offset

Register	Offset
region_attributes_2	128h

### 11.3.22.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-2. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

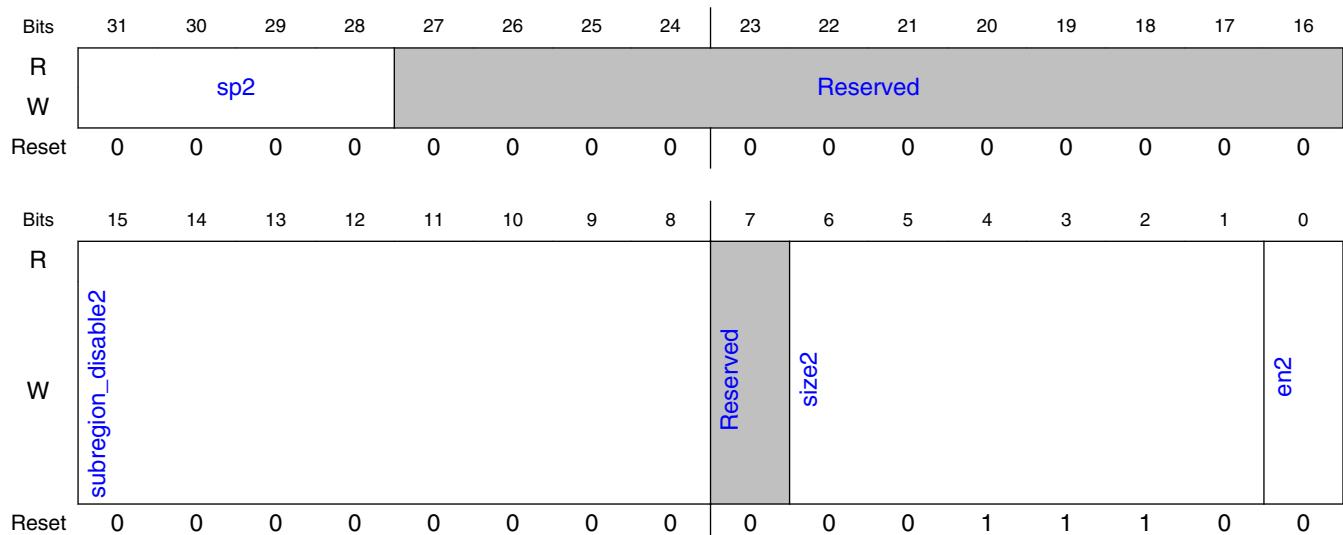
*Table continues on the next page...*

**Table 11-2. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.22.3 Diagram



### 11.3.22.4 Fields

Field	Function
31-28	sp2
sp2	

Table continues on the next page...

Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable2	<p>subregion_disable2</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size2	<p>size2</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en2	<p>en2</p> <p>Enable for region <math>n</math>:</p> <ul style="list-style-type: none"> <li>0 = region <math>n</math> is disabled</li> <li>1 = region <math>n</math> is enabled.</li> </ul>

### 11.3.23 Region Setup Low 3 Register (region\_setup\_low\_3)

#### 11.3.23.1 Offset

Register	Offset
region_setup_low_3	130h

## 11.3.23.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low3																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low3																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

## 11.3.23.3 Fields

Field	Function
31-15 base_address_low3	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.24 Region Setup High 3 Register (region\_setup\_high\_3)

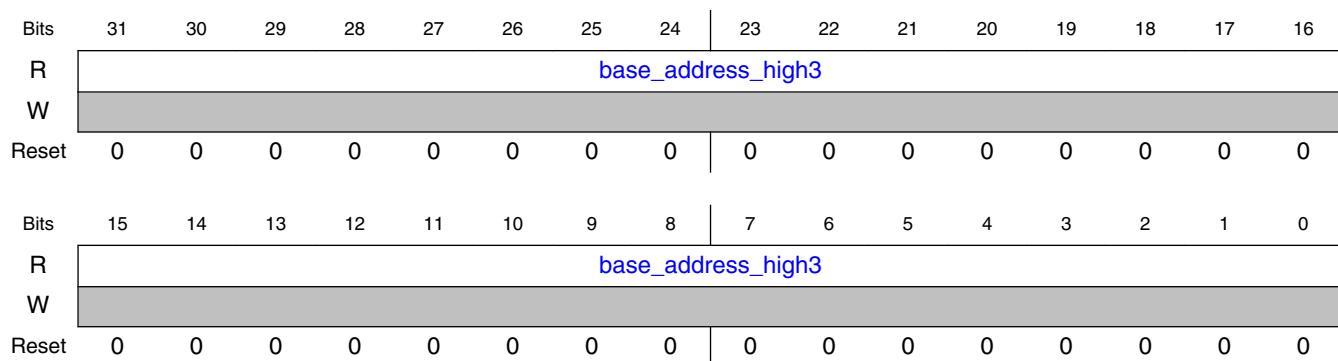
### 11.3.24.1 Offset

Register	Offset
region_setup_high_3	134h

### 11.3.24.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.24.3 Diagram



### 11.3.24.4 Fields

Field	Function
31-0 base_address_high3	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.25 Region Attributes 3 Register (region\_attributes\_3)

### 11.3.25.1 Offset

Register	Offset
region_attributes_3	138h

### 11.3.25.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-3. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-3. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	sp3								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	subregion_disable3								Reserved		size3				en3	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

### 11.3.25.4 Fields

Field	Function
31-28 sp3	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the $spn$ field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable3	Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size3	size3

Table continues on the next page...

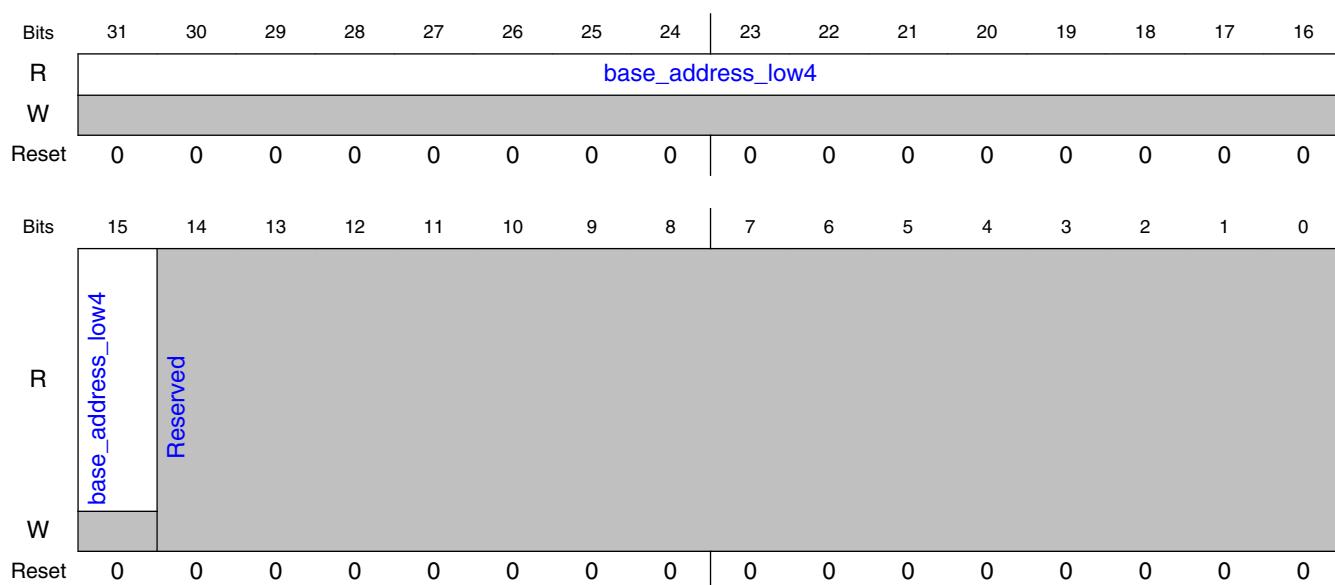
Field	Function
	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en3	<p>en3</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.26 Region Setup Low 4 Register (region\_setup\_low\_4)

### 11.3.26.1 Offset

Register	Offset
region_setup_low_4	140h

### 11.3.26.2 Diagram



### 11.3.26.3 Fields

Field	Function
31-15 base_address_low4	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.27 Region Setup High 4 Register (region\_setup\_high\_4)

#### 11.3.27.1 Offset

Register	Offset
region_setup_high_4	144h

#### 11.3.27.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.27.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high4																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high4																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.27.4 Fields

Field	Function
31-0 base_address_h igh4	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.28 Region Attributes 4 Register (region\_attributes\_4)

### 11.3.28.1 Offset

Register	Offset
region_attributes_4	148h

### 11.3.28.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-4. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

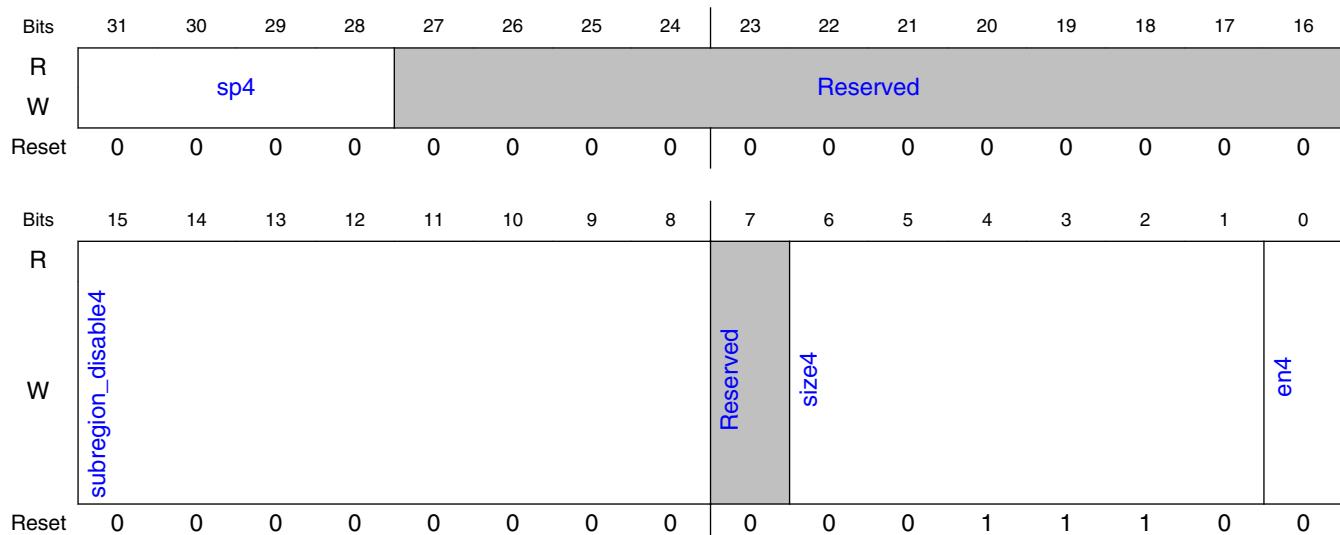
*Table continues on the next page...*

**Table 11-4. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.28.3 Diagram



### 11.3.28.4 Fields

Field	Function
31-28	sp4
sp4	

Table continues on the next page...

## Register Descriptions

Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable4	<p>subregion_disable4</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size4	<p>size4</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en4	<p>en4</p> <p>Enable for region <math>n</math>:</p> <ul style="list-style-type: none"> <li>0 = region <math>n</math> is disabled</li> <li>1 = region <math>n</math> is enabled.</li> </ul>

### 11.3.29 Region Setup Low 5 Register (region\_setup\_low\_5)

#### 11.3.29.1 Offset

Register	Offset
region_setup_low_5	150h

## 11.3.29.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low5																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low5																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

## 11.3.29.3 Fields

Field	Function
31-15 base_address_low5	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.30 Region Setup High 5 Register (region\_setup\_high\_5)

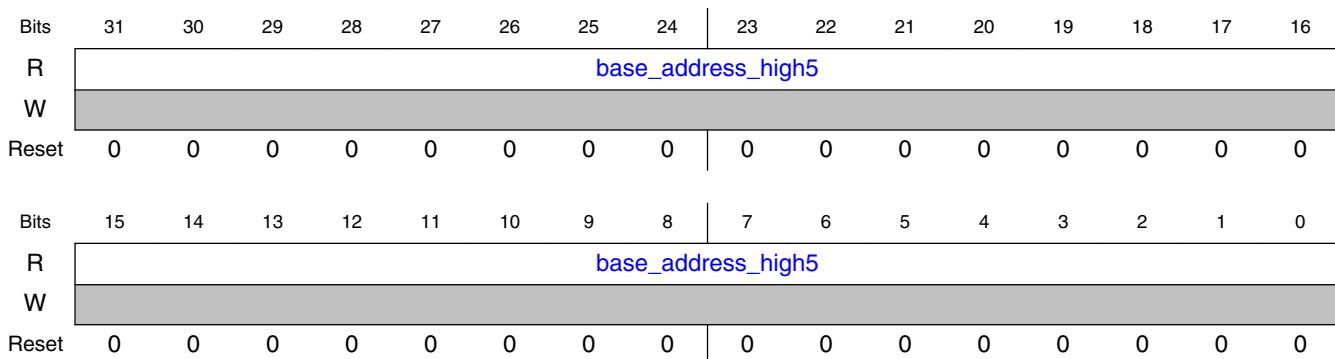
### 11.3.30.1 Offset

Register	Offset
region_setup_high_5	154h

### 11.3.30.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.30.3 Diagram



### 11.3.30.4 Fields

Field	Function
31-0 base_address_high5	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.31 Region Attributes 5 Register (region\_attributes\_5)

### 11.3.31.1 Offset

Register	Offset
region_attributes_5	158h

### 11.3.31.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-5. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-5. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.31.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16	
R	sp5									Reserved								
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0	
R	subregion_disable5									Reserved		size5				en5		
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	1	1	1	0	0	

### 11.3.31.4 Fields

Field	Function
31-28 sp5	sp5 Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable5	subregion_disable5 Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size5	size5

Table continues on the next page...

## Register Descriptions

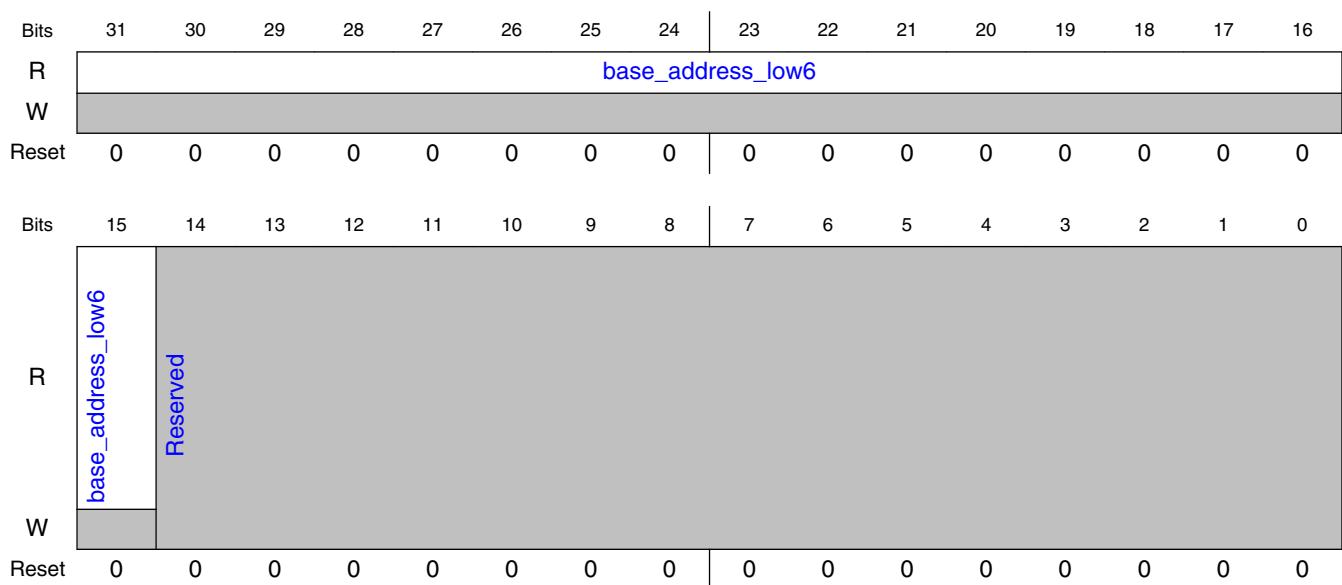
Field	Function
	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en5	<p>en5</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.32 Region Setup Low 6 Register (region\_setup\_low\_6)

### 11.3.32.1 Offset

Register	Offset
region_setup_low_6	160h

### 11.3.32.2 Diagram



### 11.3.32.3 Fields

Field	Function
31-15 base_address_low6	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.33 Region Setup High 6 Register (region\_setup\_high\_6)

#### 11.3.33.1 Offset

Register	Offset
region_setup_high_6	164h

#### 11.3.33.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.33.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high6																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high6																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.33.4 Fields

Field	Function
31-0 base_address_h igh6	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.34 Region Attributes 6 Register (region\_attributes\_6)

### 11.3.34.1 Offset

Register	Offset
region_attributes_6	168h

### 11.3.34.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-6. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

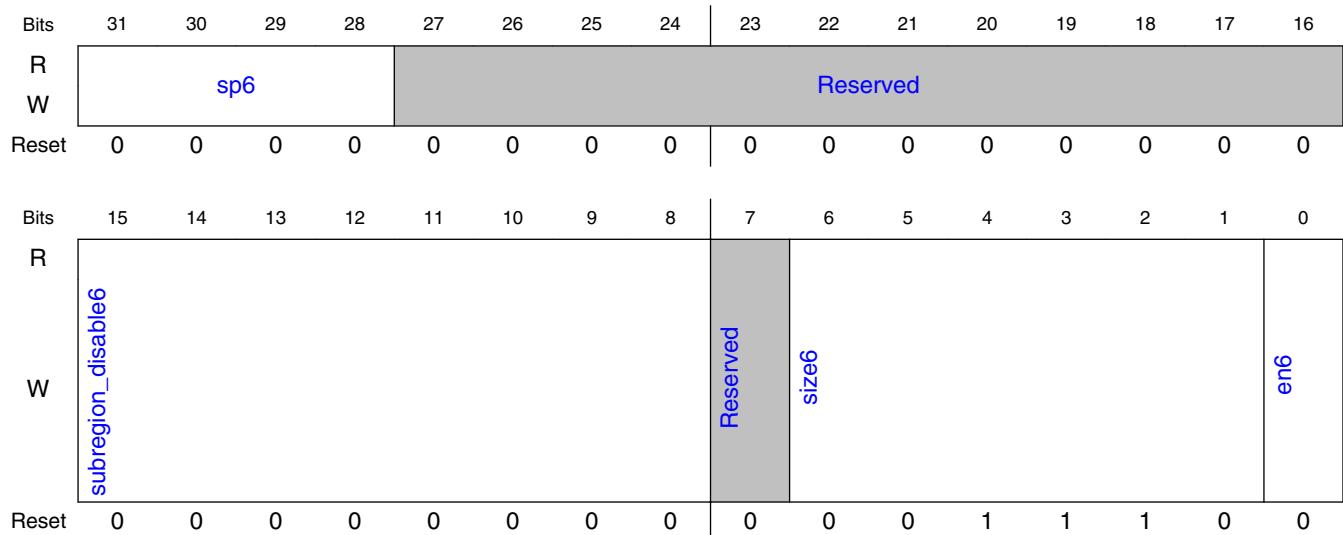
*Table continues on the next page...*

**Table 11-6. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.34.3 Diagram



### 11.3.34.4 Fields

Field	Function
31-28	sp6
sp6	

Table continues on the next page...

Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable6	<p>subregion_disable6</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size6	<p>size6</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en6	<p>en6</p> <p>Enable for region <math>n</math>:</p> <ul style="list-style-type: none"> <li>0 = region <math>n</math> is disabled</li> <li>1 = region <math>n</math> is enabled.</li> </ul>

### 11.3.35 Region Setup Low 7 Register (region\_setup\_low\_7)

#### 11.3.35.1 Offset

Register	Offset
region_setup_low_7	170h

## 11.3.35.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low7																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low7																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

## 11.3.35.3 Fields

Field	Function
31-15 base_address_low7	base_address_low7 Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.36 Region Setup High 7 Register (region\_setup\_high\_7)

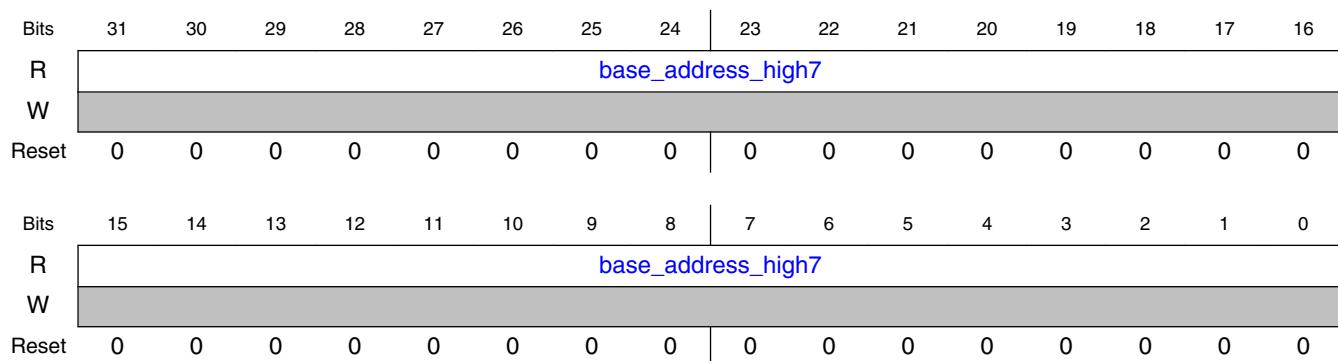
### 11.3.36.1 Offset

Register	Offset
region_setup_high_7	174h

### 11.3.36.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.36.3 Diagram



### 11.3.36.4 Fields

Field	Function
31-0 base_address_high7	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.37 Region Attributes 7 Register (region\_attributes\_7)

### 11.3.37.1 Offset

Register	Offset
region_attributes_7	178h

### 11.3.37.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-7. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-7. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.37.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	sp7								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	subregion_disable7								Reserved		size7				en7	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

### 11.3.37.4 Fields

Field	Function
31-28 sp7	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the <i>spn</i> field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable7	Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size7	size7

Table continues on the next page...

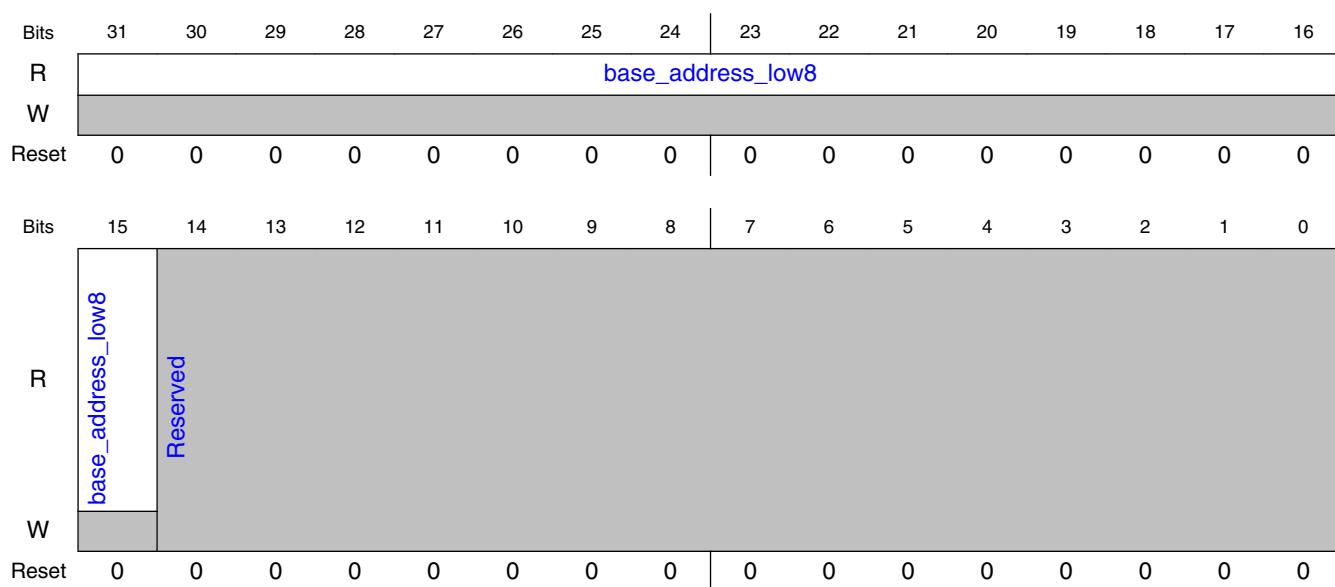
Field	Function
	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en7	<p>en7</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.38 Region Setup Low 8 Register (region\_setup\_low\_8)

### 11.3.38.1 Offset

Register	Offset
region_setup_low_8	180h

### 11.3.38.2 Diagram



### 11.3.38.3 Fields

Field	Function
31-15 base_address_low8	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.39 Region Setup High 8 Register (region\_setup\_high\_8)

#### 11.3.39.1 Offset

Register	Offset
region_setup_high_8	184h

#### 11.3.39.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.39.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high8																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high8																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.39.4 Fields

Field	Function
31-0 base_address_h igh8	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.40 Region Attributes 8 Register (region\_attributes\_8)

### 11.3.40.1 Offset

Register	Offset
region_attributes_8	188h

### 11.3.40.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-8. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

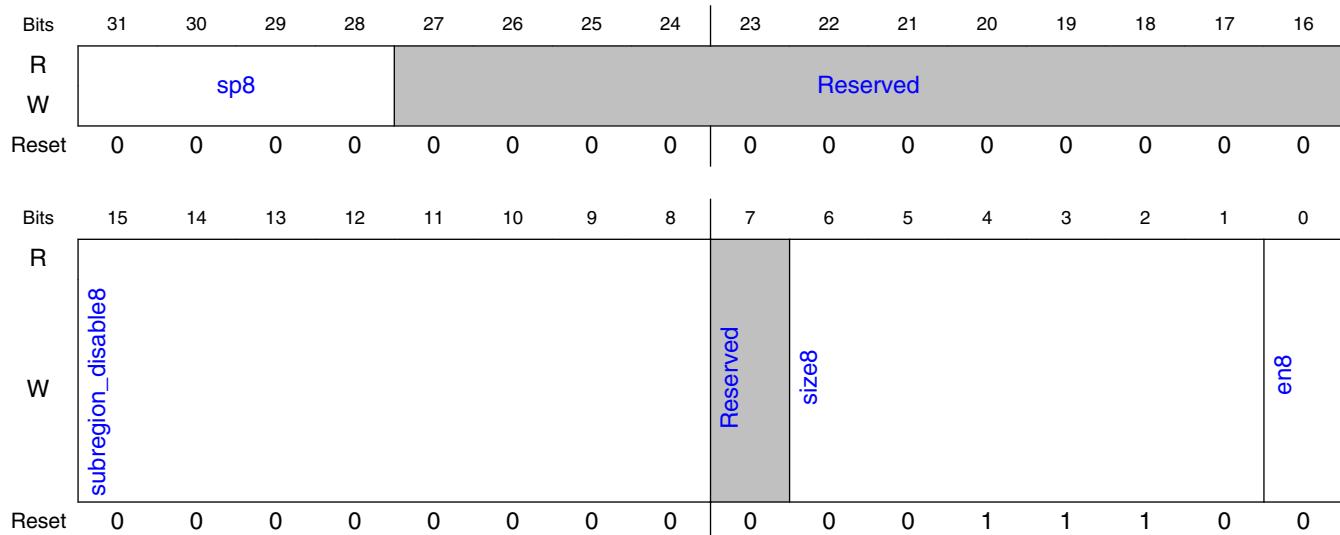
*Table continues on the next page...*

**Table 11-8. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.40.3 Diagram



### 11.3.40.4 Fields

Field	Function
31-28	sp8
sp8	

Table continues on the next page...

## Register Descriptions

Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the <i>spn</i> field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable8	<p>subregion_disable8</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size8	<p>size8</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en8	<p>en8</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

### 11.3.41 Region Setup Low 9 Register (region\_setup\_low\_9)

#### 11.3.41.1 Offset

Register	Offset
region_setup_low_9	190h

### 11.3.41.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low9																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low9																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.41.3 Fields

Field	Function
31-15 base_address_low9	base_address_low9 Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.42 Region Setup High 9 Register (region\_setup\_high\_9)

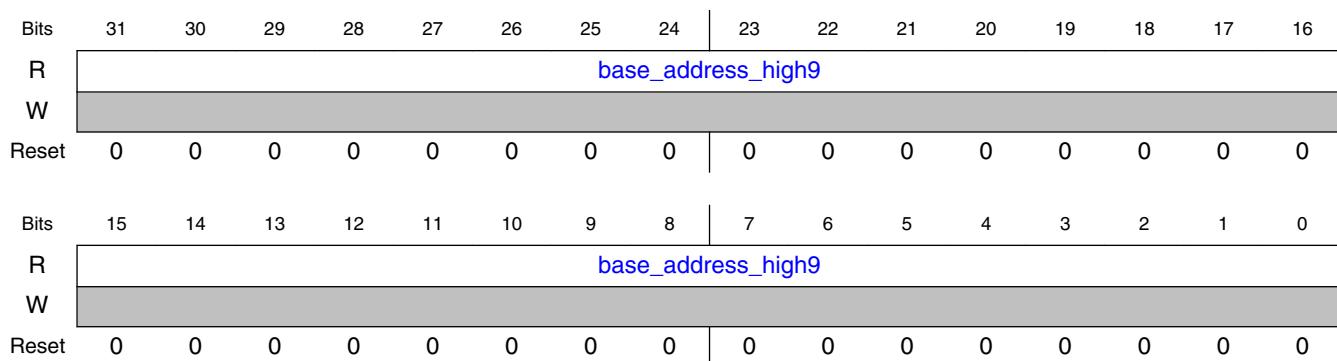
### 11.3.42.1 Offset

Register	Offset
region_setup_high_9	194h

### 11.3.42.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.42.3 Diagram



### 11.3.42.4 Fields

Field	Function
31-0 base_address_high9	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.43 Region Attributes 9 Register (region\_attributes\_9)

### 11.3.43.1 Offset

Register	Offset
region_attributes_9	198h

### 11.3.43.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-9. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-9. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.43.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16	
R	sp9									Reserved								
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0	
R	subregion_disable9									Reserved		size9				en9		
W																		
Reset	0	0	0	0	0	0	0	0		0	0	0	1	1	1	0	0	

### 11.3.43.4 Fields

Field	Function
31-28 sp9	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the $spn$ field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable9	Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size9	size9

Table continues on the next page...

## Register Descriptions

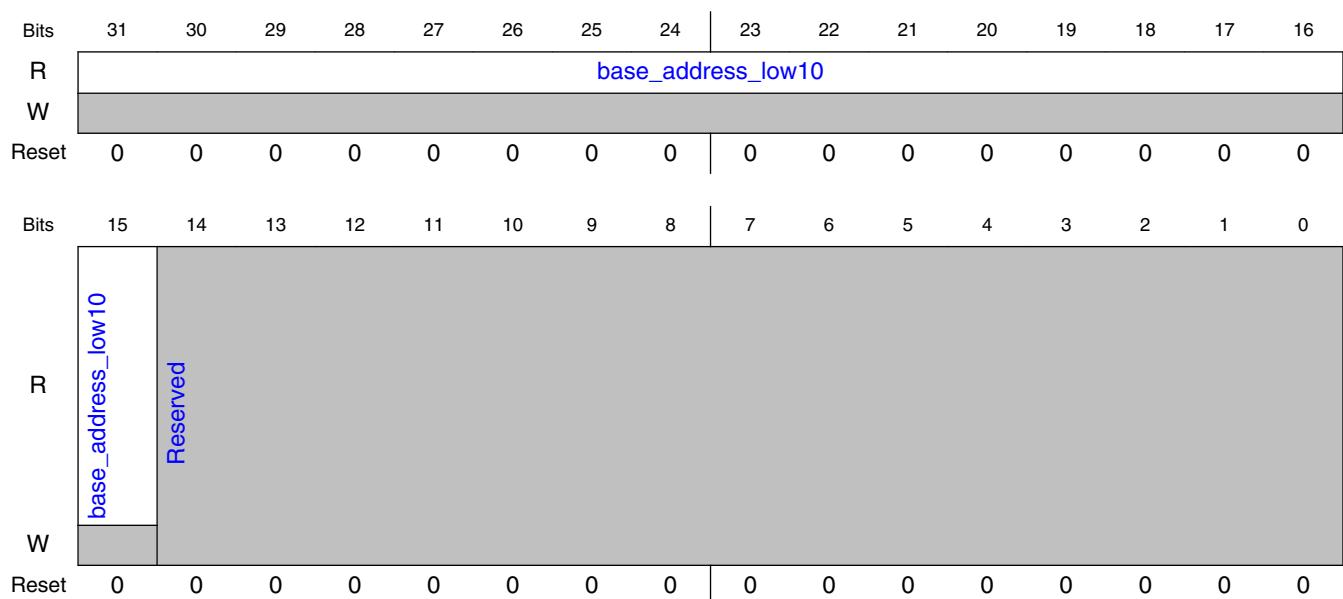
Field	Function
	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en9	<p>en9</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.44 Region Setup Low 10 Register (region\_setup\_low\_10)

### 11.3.44.1 Offset

Register	Offset
region_setup_low_10	1A0h

### 11.3.44.2 Diagram



### 11.3.44.3 Fields

Field	Function
31-15 base_address_low10	<p>base_address_low10</p> <p>Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.</p> <p>The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:</p> <ul style="list-style-type: none"> <li>b00100000000000000000</li> <li>b01000000000000000000</li> <li>b01100000000000000000</li> <li>b10000000000000000000</li> <li>b10100000000000000000</li> <li>b11000000000000000000</li> <li>b11100000000000000000</li> </ul> <p>If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.</p>
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.45 Region Setup High 10 Register (region\_setup\_high\_10)

#### 11.3.45.1 Offset

Register	Offset
region_setup_high_10	1A4h

#### 11.3.45.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.45.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high10																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high10																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.45.4 Fields

Field	Function
31-0 base_address_h igh10	base_address_high10 Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0. The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.46 Region Attributes 10 Register (region\_attributes\_10)

### 11.3.46.1 Offset

Register	Offset
region_attributes_10	1A8h

### 11.3.46.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-10. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

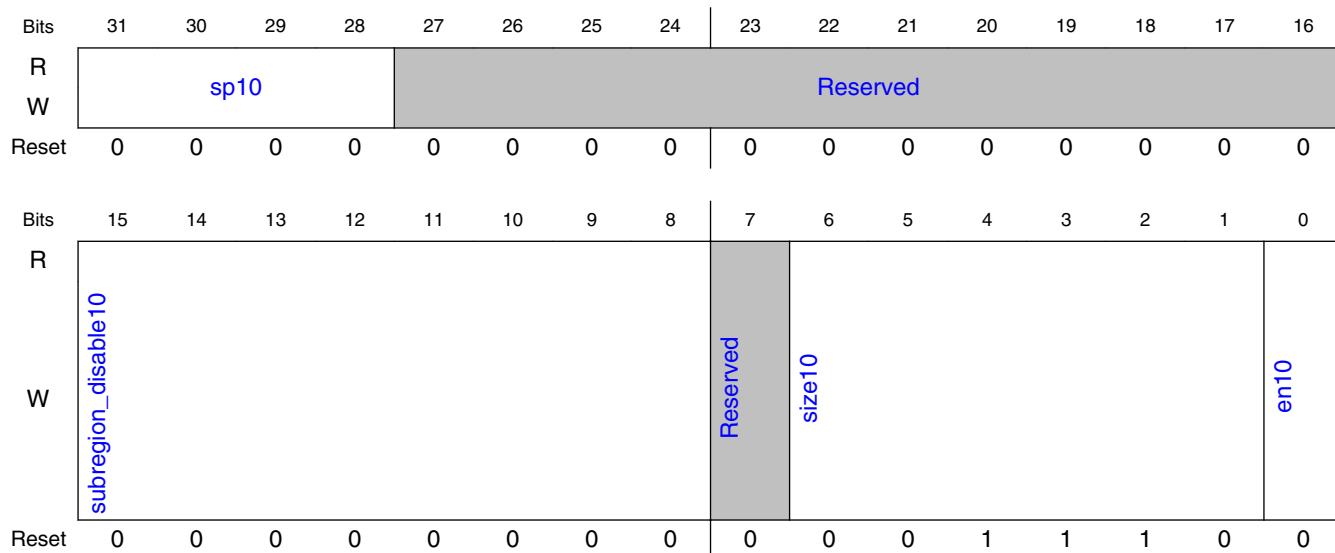
*Table continues on the next page...*

**Table 11-10. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.46.3 Diagram



### 11.3.46.4 Fields

Field	Function
31-28	sp10
sp10	

Table continues on the next page...

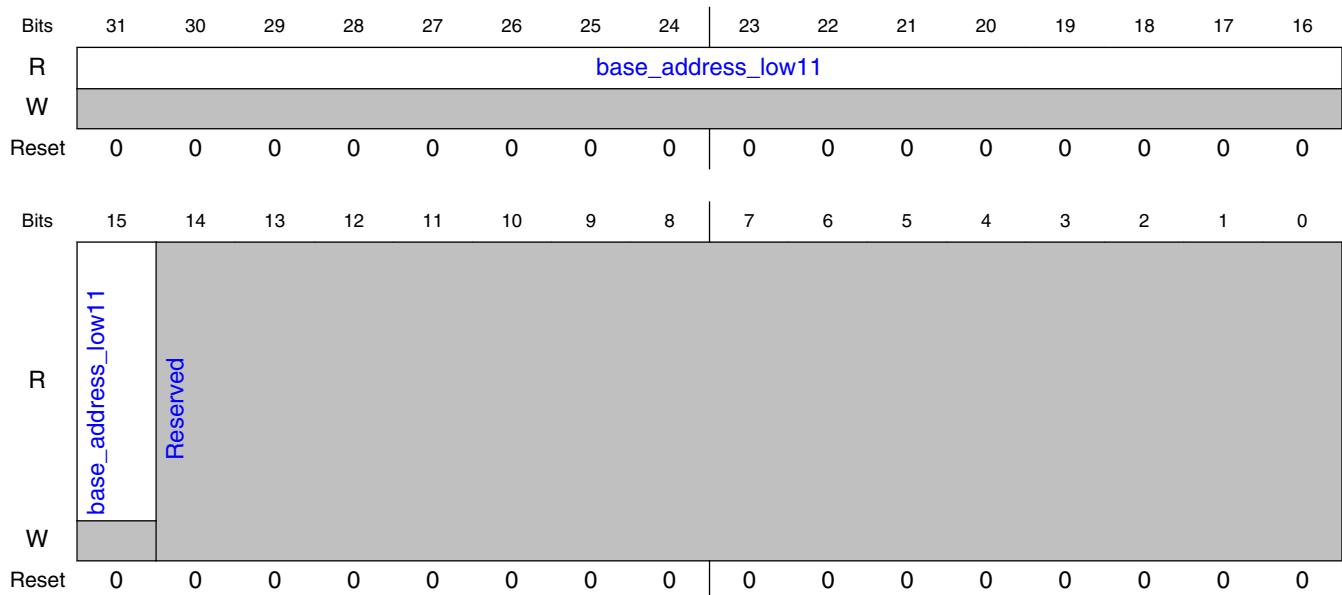
Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable10	<p>subregion_disable10</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size10	<p>size10</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en10	<p>en10</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

### 11.3.47 Region Setup Low 11 Register (region\_setup\_low\_11)

#### 11.3.47.1 Offset

Register	Offset
region_setup_low_11	1B0h

## 11.3.47.2 Diagram



## 11.3.47.3 Fields

Field	Function
31-15 base_address_low11	<p>base_address_low11</p> <p>Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.</p> <p>The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:</p> <ul style="list-style-type: none"> <li>b001000000000000000000000</li> <li>b010000000000000000000000</li> <li>b011000000000000000000000</li> <li>b100000000000000000000000</li> <li>b101000000000000000000000</li> <li>b110000000000000000000000</li> <li>b111000000000000000000000</li> </ul> <p>If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.</p>
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.48 Region Setup High 11 Register (region\_setup\_high\_11)

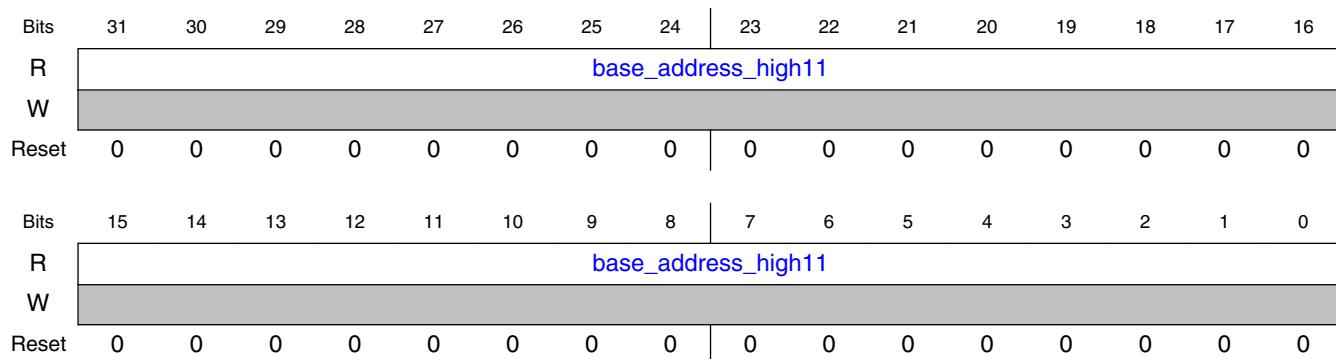
### 11.3.48.1 Offset

Register	Offset
region_setup_high_11	1B4h

### 11.3.48.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.48.3 Diagram



### 11.3.48.4 Fields

Field	Function
31-0 base_address_high11	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.49 Region Attributes 11 Register (region\_attributes\_11)

### 11.3.49.1 Offset

Register	Offset
region_attributes_11	1B8h

### 11.3.49.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-11. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-11. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.49.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	sp11								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved		size11					
W	subregion_disable11								0	0	0	1	1	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

### 11.3.49.4 Fields

Field	Function
31-28 sp11	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable11	subregion_disable11 Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size11	

Table continues on the next page...

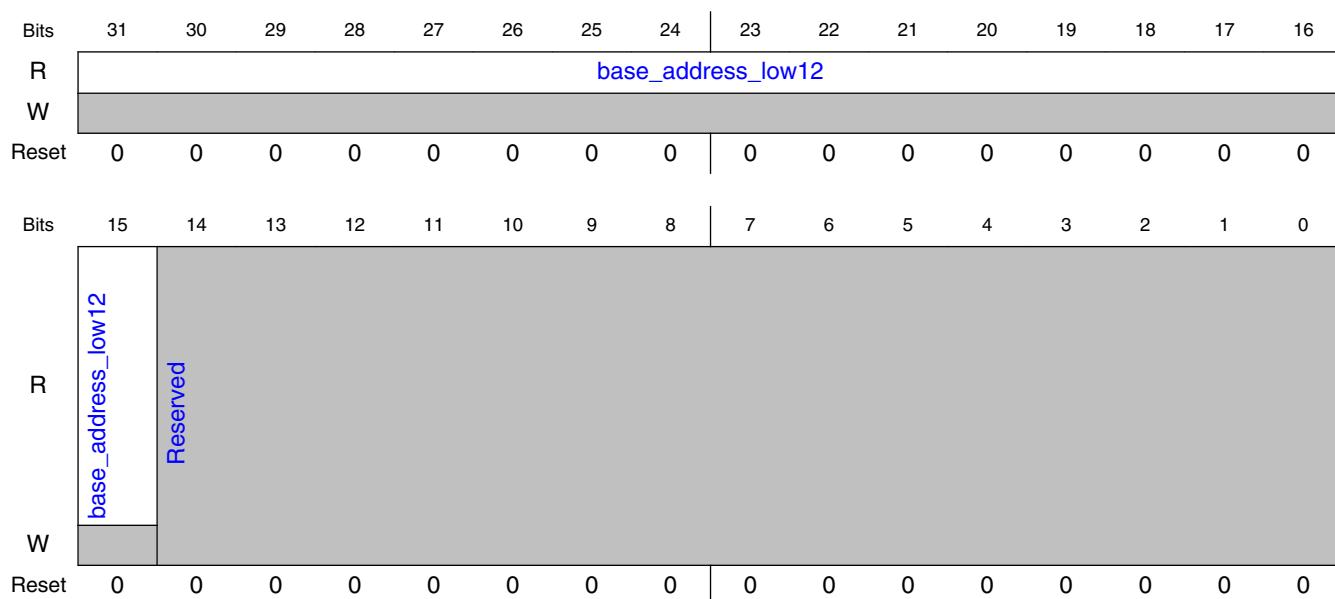
Field	Function
size11	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en11	<p>en11</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.50 Region Setup Low 12 Register (region\_setup\_low\_12)

### 11.3.50.1 Offset

Register	Offset
region_setup_low_12	1C0h

### 11.3.50.2 Diagram



### 11.3.50.3 Fields

Field	Function
31-15 base_address_low12	<p>Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.</p> <p>The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:</p> <ul style="list-style-type: none"> <li>b00100000000000000000</li> <li>b01000000000000000000</li> <li>b01100000000000000000</li> <li>b10000000000000000000</li> <li>b10100000000000000000</li> <li>b11000000000000000000</li> <li>b11100000000000000000</li> </ul> <p>If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.</p>
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.51 Region Setup High 12 Register (region\_setup\_high\_12)

### 11.3.51.1 Offset

Register	Offset
region_setup_high_12	1C4h

### 11.3.51.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.51.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high12																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high12																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.51.4 Fields

Field	Function
31-0 base_address_h igh12	base_address_high12  Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.52 Region Attributes 12 Register (region\_attributes\_12)

### 11.3.52.1 Offset

Register	Offset
region_attributes_12	1C8h

### 11.3.52.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-12. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

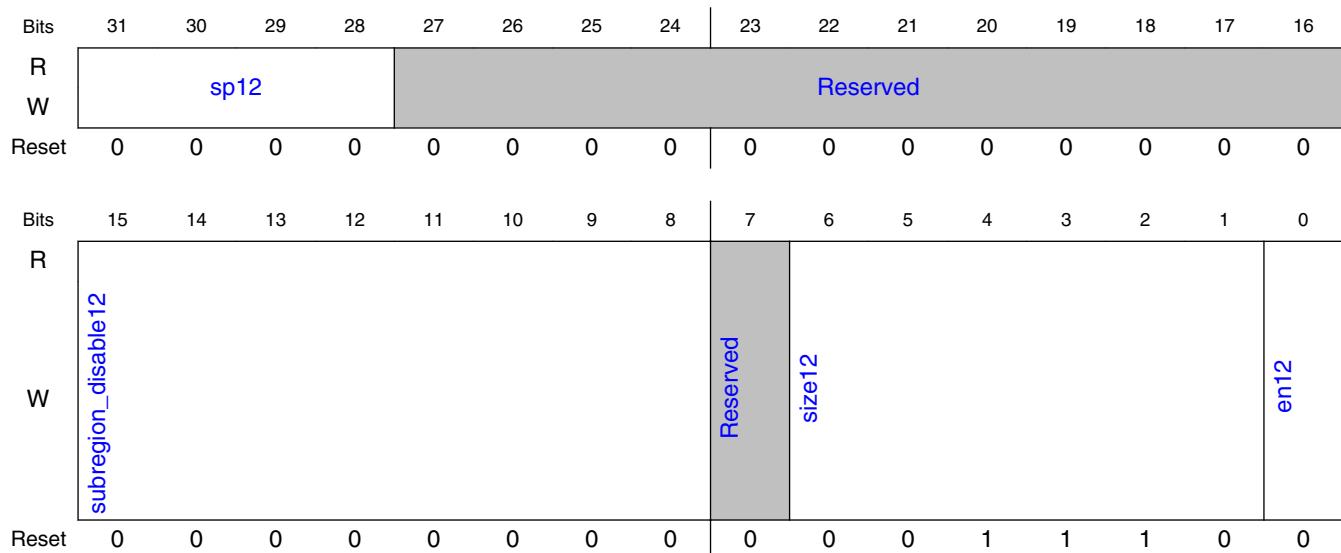
*Table continues on the next page...*

**Table 11-12. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.52.3 Diagram



### 11.3.52.4 Fields

Field	Function
31-28	sp12
sp12	

*Table continues on the next page...*

## Register Descriptions

Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable12	<p>subregion_disable12</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size12	<p>size12</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en12	<p>en12</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.53 Region Setup Low 13 Register (region\_setup\_low\_13)

### 11.3.53.1 Offset

Register	Offset
region_setup_low_13	1D0h

### 11.3.53.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_low13																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_low13																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.53.3 Fields

Field	Function
31-15 base_address_low13	Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:  b00100000000000000000 b01000000000000000000 b01100000000000000000 b10000000000000000000 b10100000000000000000 b11000000000000000000 b11100000000000000000  If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.54 Region Setup High 13 Register (region\_setup\_high\_13)

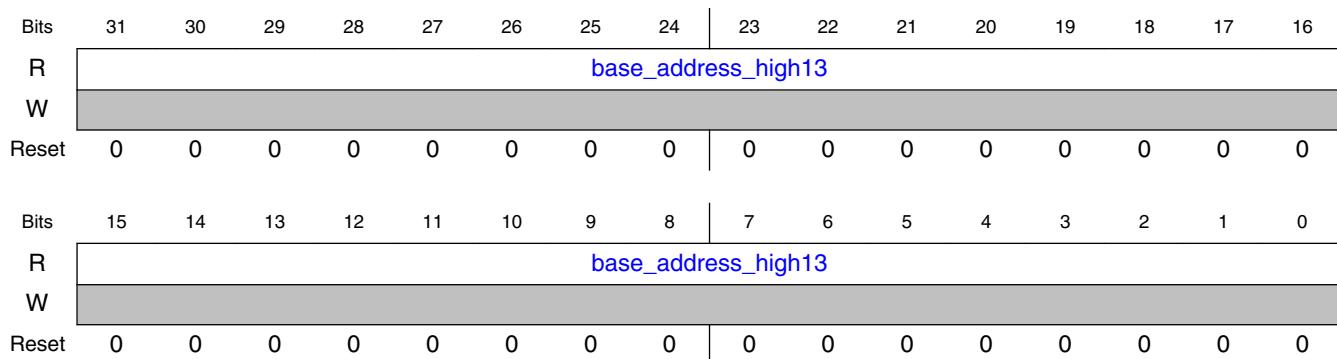
### 11.3.54.1 Offset

Register	Offset
region_setup_high_13	1D4h

### 11.3.54.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.54.3 Diagram



### 11.3.54.4 Fields

Field	Function
31-0 base_address_high13	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.55 Region Attributes 13 Register (region\_attributes\_13)

### 11.3.55.1 Offset

Register	Offset
region_attributes_13	1D8h

### 11.3.55.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-13. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

Table continues on the next page...

**Table 11-13. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.55.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	sp13								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	subregion_disable13								Reserved		size13				en13	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

### 11.3.55.4 Fields

Field	Function
31-28 sp13	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable13	subregion_disable13 Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size13	

Table continues on the next page...

## Register Descriptions

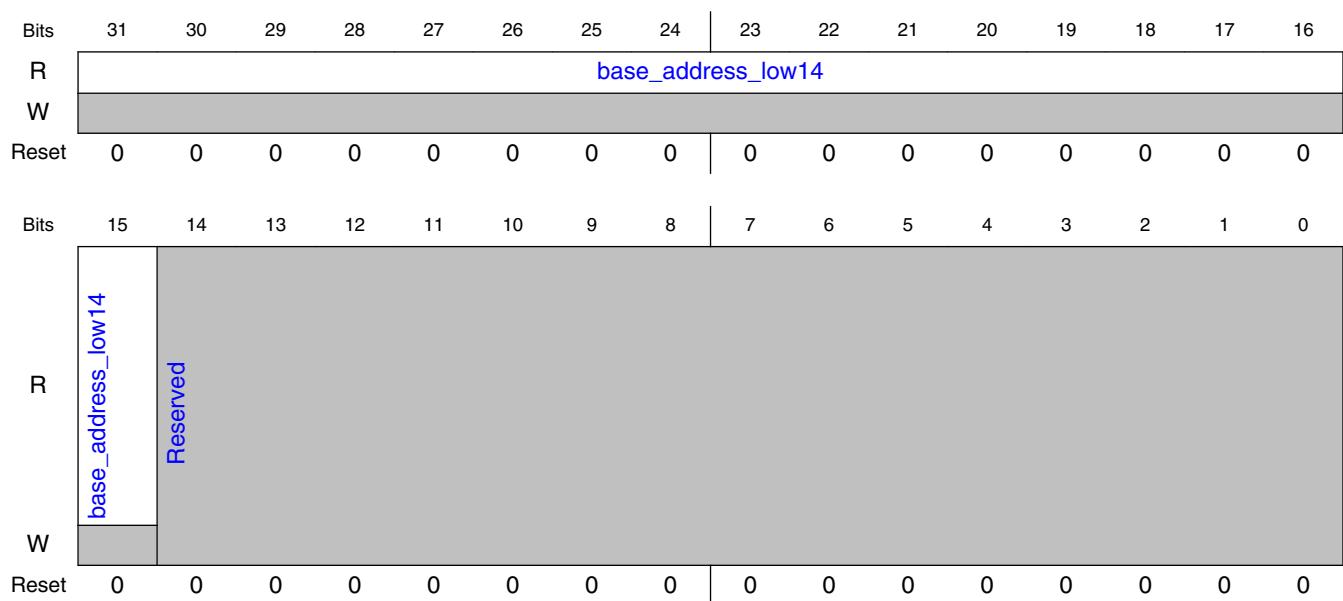
Field	Function
size13	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en13	<p>en13</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.56 Region Setup Low 14 Register (region\_setup\_low\_14)

### 11.3.56.1 Offset

Register	Offset
region_setup_low_14	1E0h

### 11.3.56.2 Diagram



### 11.3.56.3 Fields

Field	Function
31-15 base_address_low14	<p>base_address_low14</p> <p>Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.</p> <p>The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:</p> <ul style="list-style-type: none"> <li>b00100000000000000000</li> <li>b01000000000000000000</li> <li>b01100000000000000000</li> <li>b10000000000000000000</li> <li>b10100000000000000000</li> <li>b11000000000000000000</li> <li>b11100000000000000000</li> </ul> <p>If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.</p>
14-0 —	Reserved, Should be Zero (SBZ).

### 11.3.57 Region Setup High 14 Register (region\_setup\_high\_14)

#### 11.3.57.1 Offset

Register	Offset
region_setup_high_14	1E4h

#### 11.3.57.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.57.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	base_address_high14																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	base_address_high14																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 11.3.57.4 Fields

Field	Function
31-0 base_address_h igh14	base_address_high14 Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0. The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.58 Region Attributes 14 Register (region\_attributes\_14)

### 11.3.58.1 Offset

Register	Offset
region_attributes_14	1E8h

### 11.3.58.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-14. Region size**

<b>size&lt;n&gt; field</b>	<b>Size of region &lt;n&gt;</b>	<b>Base address [1] constraints</b>
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero
b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero

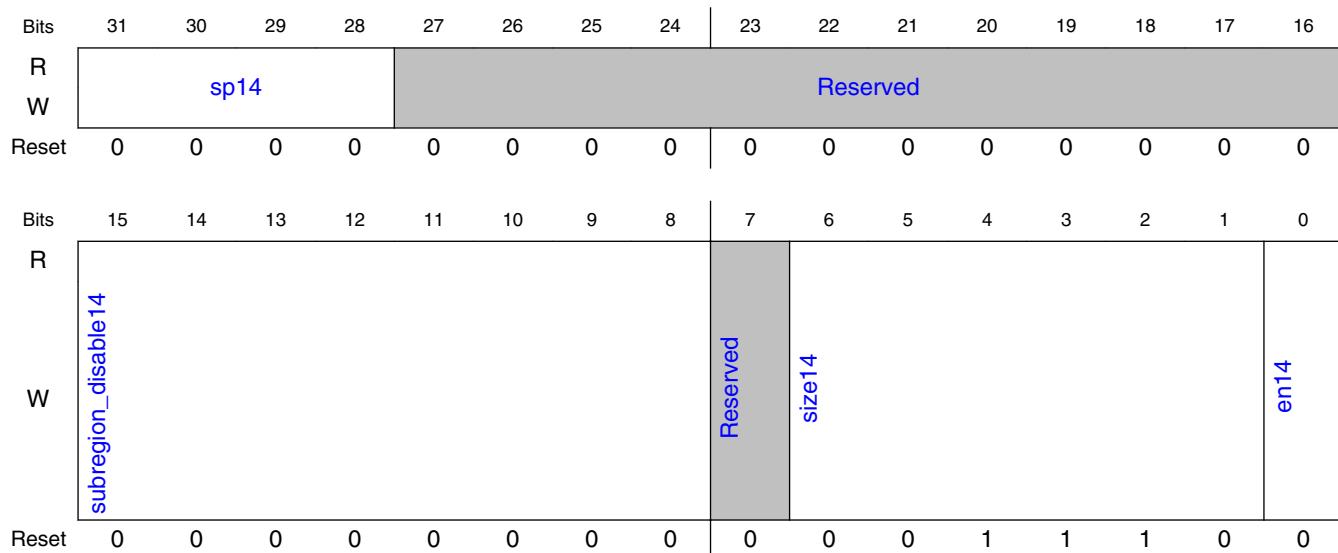
*Table continues on the next page...*

**Table 11-14. Region size (continued)**

b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.58.3 Diagram



### 11.3.58.4 Fields

Field	Function
31-28	sp14
sp14	

Table continues on the next page...

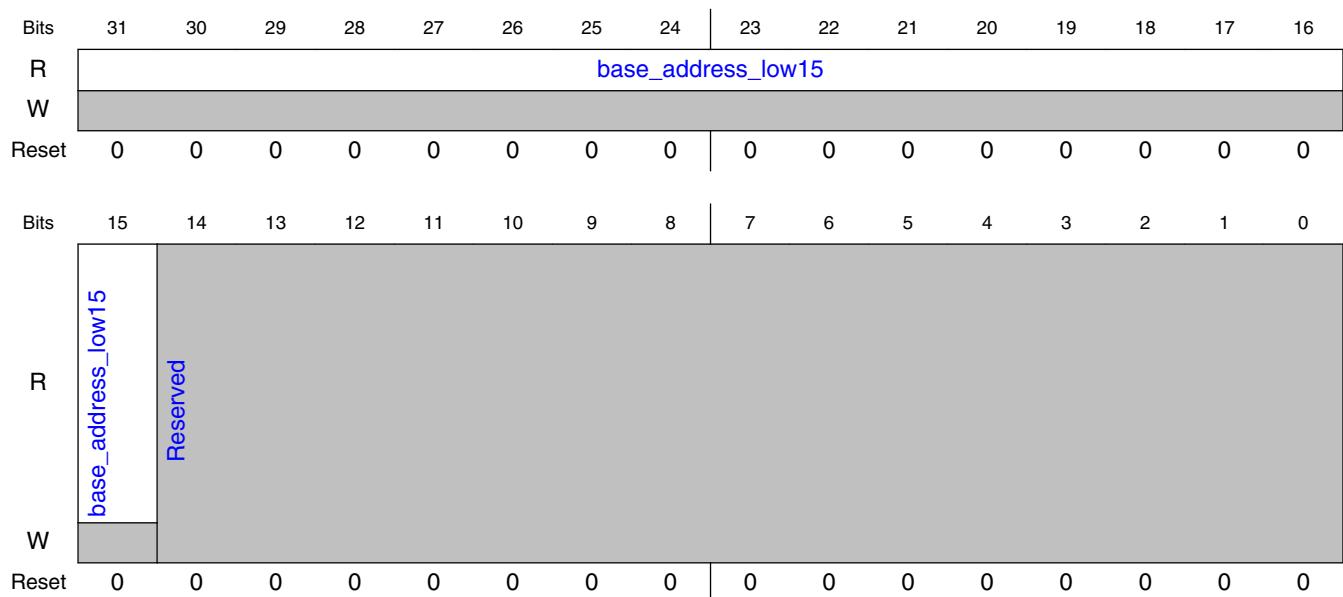
Field	Function
	Permissions setting for region $n$ . If an AXI transaction occurs to region $n$ , the value in the $spn$ field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable14	<p>subregion_disable14</p> <p>Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled:</p> <ul style="list-style-type: none"> <li>Bit [15] = 1 Subregion 7 is disabled.</li> <li>Bit [14] = 1 Subregion 6 is disabled.</li> <li>Bit [13] = 1 Subregion 5 is disabled.</li> <li>Bit [12] = 1 Subregion 4 is disabled.</li> <li>Bit [11] = 1 Subregion 3 is disabled.</li> <li>Bit [10] = 1 Subregion 2 is disabled.</li> <li>Bit [9] = 1 Subregion 1 is disabled.</li> <li>Bit [8] = 1 Subregion 0 is disabled.</li> </ul>
7 —	Reserved, Should be Zero (SBZ).
6-1 size14	<p>size14</p> <p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en14	<p>en14</p> <p>Enable for region <math>n</math>:</p> <ul style="list-style-type: none"> <li>0 = region <math>n</math> is disabled</li> <li>1 = region <math>n</math> is enabled.</li> </ul>

## 11.3.59 Region Setup Low 15 Register (region\_setup\_low\_15)

### 11.3.59.1 Offset

Register	Offset
region_setup_low_15	1F0h

## 11.3.59.2 Diagram



## 11.3.59.3 Fields

Field	Function
31-15 base_address_low15	<p>base_address_low15</p> <p>Controls the base address [31:15] of region n. For region 0, this field is Read Only (RO). The TZASC sets the base address of region 0 to 0x0.</p> <p>The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. For example, if the size of a region is 512MB, and it is not at address 0x0, the only valid settings for this field are:</p> <ul style="list-style-type: none"> <li>b001000000000000000000000</li> <li>b010000000000000000000000</li> <li>b011000000000000000000000</li> <li>b100000000000000000000000</li> <li>b101000000000000000000000</li> <li>b110000000000000000000000</li> <li>b111000000000000000000000</li> </ul> <p>If you attempt to set an inappropriate base address for the size of the region, the TZASC ignores certain bits depending on the region size. See Table on register size for more information.</p>
14-0 —	Reserved, Should be Zero (SBZ).

## 11.3.60 Region Setup High 15 Register (region\_setup\_high\_15)

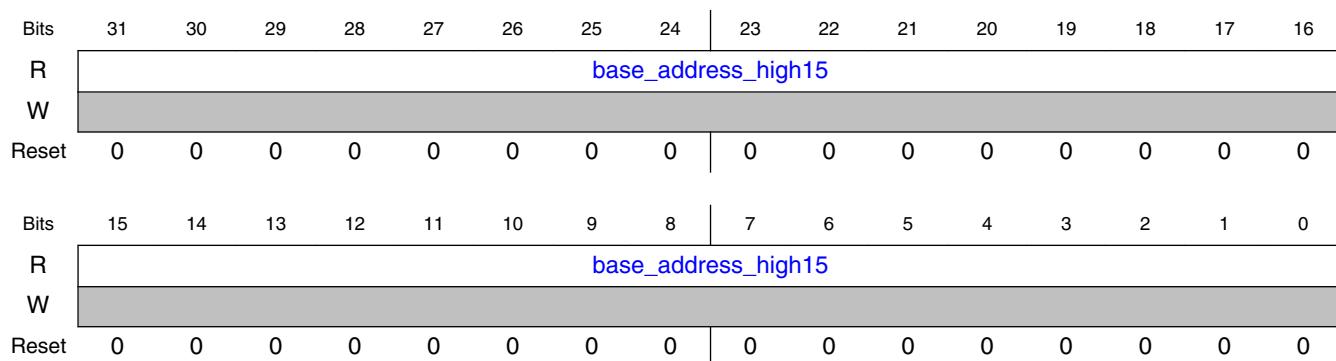
### 11.3.60.1 Offset

Register	Offset
region_setup_high_15	1F4h

### 11.3.60.2 Function

The region\_setup\_high register is none for region 0, for other regions it controls the base address [63:32] of region n. There are no usage constraints. Available in all configurations of the TZASC.

### 11.3.60.3 Diagram



### 11.3.60.4 Fields

Field	Function
31-0 base_address_high15	Controls the base address [63:32] of region n. For region 0, this field is RO. The TZASC sets the base address of region 0 to 0x0.  The TZASC only permits a region to start at address 0x0, or at a multiple of its region size. If you program a region size to be 8GB or more, then the TZASC might ignore certain bits depending on the region size.

## 11.3.61 Region Attributes 15 Register (region\_attributes\_15)

### 11.3.61.1 Offset

Register	Offset
region_attributes_15	1F8h

### 11.3.61.2 Function

The region\_attributes\_n register controls the permissions, region size, subregion disable, and region enable. There are no usage constraints. Available in all configurations of the TZASC.

**Table 11-15. Region size**

size<n> field	Size of region <n>	Base address [1] constraints
b000000-b001101	Reserved	-
b001110	32KB	-
b001111	64KB	Bit [15] must be zero
b010000	128KB	Bits [16:15] must be zero
b010001	256KB	Bits [17:15] must be zero
b010010	512KB	Bits [18:15] must be zero
b010011	1MB	Bits [19:15] must be zero
b010100	2MB	Bits [20:15] must be zero
b010101	4MB	Bits [21:15] must be zero
b010110	8MB	Bits [22:15] must be zero
b010111	16MB	Bits [23:15] must be zero
b011000	32MB	Bits [24:15] must be zero
b011001	64MB	Bits [25:15] must be zero
b011010	128MB	Bits [26:15] must be zero
b011011	256MB	Bits [27:15] must be zero
b011100	512MB	Bits [28:15] must be zero
b011101	1GB	Bits [29:15] must be zero
b011110	2GB	Bits [30:15] must be zero
b011111	4GB	Bits [31:15] must be zero
b100000	8GB	Bits [32:15] must be zero
b100001	16GB	Bits [33:15] must be zero
b100010	32GB	Bits [34:15] must be zero
b100011	64GB	Bits [35:15] must be zero

*Table continues on the next page...*

**Table 11-15. Region size (continued)**

b100100	128GB	Bits [36:15] must be zero
b100101	256GB	Bits [37:15] must be zero
b100110	512GB	Bits [38:15] must be zero
b100111	1TB	Bits [39:15] must be zero
b101000	2TB	Bits [40:15] must be zero
b101001	4TB	Bits [41:15] must be zero
b101010	8TB	Bits [42:15] must be zero
b101011	16TB	Bits [43:15] must be zero
b101100	32TB	Bits [44:15] must be zero
b101101	64TB	Bits [45:15] must be zero
b101110	128TB	Bits [46:15] must be zero
b101111	256TB	Bits [47:15] must be zero
b110000	512TB	Bits [48:15] must be zero
b110001	1PB	Bits [49:15] must be zero
b110010	2PB	Bits [50:15] must be zero
b110011	4PB	Bits [51:15] must be zero
b110100	8PB	Bits [52:15] must be zero
b110101	16PB	Bits [53:15] must be zero
b110110	32PB	Bits [54:15] must be zero
b110111	64PB	Bits [55:15] must be zero
b111000	128PB	Bits [56:15] must be zero
b111001	256PB	Bits [57:15] must be zero
b111010	512PB	Bits [58:15] must be zero
b111011	1EB	Bits [59:15] must be zero
b111100	2EB	Bits [60:15] must be zero
b111101	4EB	Bits [61:15] must be zero
b111110	8EB	Bits [62:15] must be zero
b111111	16EB	Bits [63:15] must be zero

[1]The region\_setup\_low\_<n> Register and region\_setup\_high\_<n> Register contain the base address. See Table 313 on page 318 and Table 314 on page 319.

### 11.3.61.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	sp15								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved		size15					
W	subregion_disable15								0	0	0	1	1	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

### 11.3.61.4 Fields

Field	Function
31-28 sp15	Permissions setting for region <i>n</i> . If an AXI transaction occurs to region <i>n</i> , the value in the spn field controls whether the TZASC permits the transaction to proceed.
27-16 —	Reserved, Should be Zero (SBZ).
15-8 subregion_disable15	subregion_disable15 Regions are split into eight equal-sized sub-regions, and each bit enables the corresponding subregion to be disabled: Bit [15] = 1 Subregion 7 is disabled. Bit [14] = 1 Subregion 6 is disabled. Bit [13] = 1 Subregion 5 is disabled. Bit [12] = 1 Subregion 4 is disabled. Bit [11] = 1 Subregion 3 is disabled. Bit [10] = 1 Subregion 2 is disabled. Bit [9] = 1 Subregion 1 is disabled. Bit [8] = 1 Subregion 0 is disabled.
7 —	Reserved, Should be Zero (SBZ).
6-1 size15	size15

Table continues on the next page...

Field	Function
size15	<p>Size of region <math>n</math>. The AXI address width, that is AXI_ADDRESS_MSB+1, controls the upper limit value of this field.</p> <p>The table below shows how the size <math>n</math> field controls the region size and what constraints, if any, the TZASC applies to the base address to ensure that a region starts on the boundary of the region size.</p>
0 en15	<p>en15</p> <p>Enable for region <math>n</math>:</p> <p>0 = region <math>n</math> is disabled</p> <p>1 = region <math>n</math> is enabled.</p>

## 11.3.62 Integration Test Control Register (itcrg)

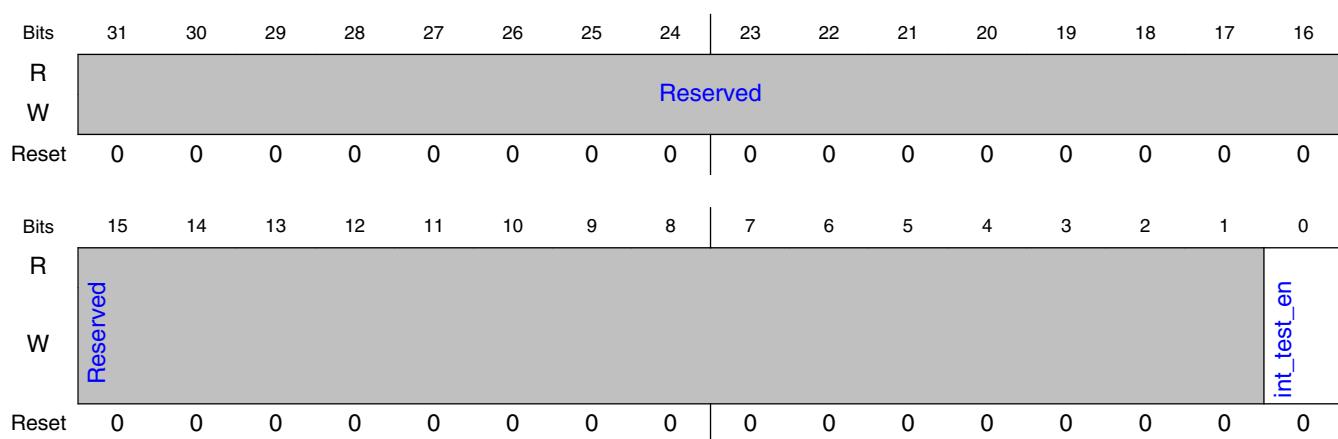
### 11.3.62.1 Offset

Register	Offset
itcrg	E00h

### 11.3.62.2 Function

The itcrg register enables the integration test logic. Use this in integration test mode. Available in all configurations of the TZASC.

### 11.3.62.3 Diagram



### 11.3.62.4 Fields

Field	Function
31-1 —	Undefined. Write as zero.
0 int_test_en	int_test_en Controls the enabling of, or provides the status of, the integration test logic: 0 = integration test logic is disabled 1 = integration test logic is enabled.

## 11.3.63 Integration Test Input Register (itip)

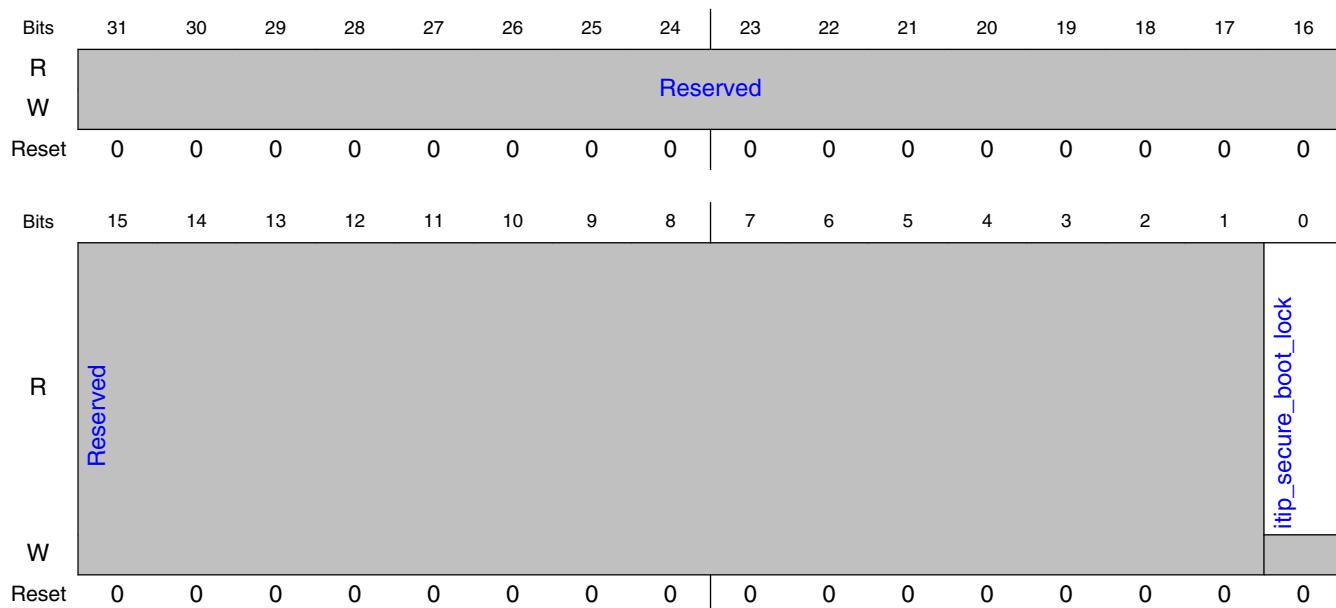
### 11.3.63.1 Offset

Register	Offset
itip	E04h

### 11.3.63.2 Function

The itip register enables a processor to read the status of secure\_boot\_lock. Integration test logic must be enabled otherwise reads return 0x0. See Integration Test Control Register for information about enabling the integration test logic. Available in all configurations of the TZASC.

### 11.3.63.3 Diagram



### 11.3.63.4 Fields

Field	Function
31-1 —	Reserved
0 itip_secure_boot_lock	itip_secure_boot_lock

### 11.3.64 Integration Test Output Register (itop)

#### 11.3.64.1 Offset

Register	Offset
itop	E08h

## 11.3.64.2 Function

The itop register enables a processor to set the status of tzasc\_int in integration test mode. Usage constraints Integration test logic must be enabled otherwise it ignores writes and reads return 0x0. See Integration Test Control Register for information about enabling the integration test logic. Available in all configurations of the TZASC.

## 11.3.64.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 11.3.64.4 Fields

Field	Function
31-1	Undefined. Write as zero.
—	
0	itop_int
itop_int	Set or reset the value of tzasc_int port by writing 1 or 0 into itop_int bit. If you read, the written value can be read back. 0 = tzasc_int is LOW 1 = tzasc_int is HIGH.

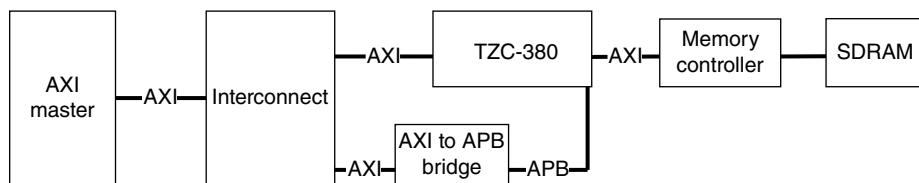
## 11.4 Functional description

This chapter describes the TZASC operation. It contains the following sections:

- [Functional operation](#)
- [Constraints of use](#)

## 11.4.1 Functional operation

The TZASC performs security checks on AXI accesses to memory and DDR memory space. This supports configurable number of regions. Each region is programmable for size, base address, enable, and security parameters. Using the secure\_boot\_lock, the programmers view can be locked to prevent erroneous writes. See [Preventing writes to registers and using secure\\_boot\\_lock](#). The TZASC provides programmability in reporting faults using AXI response channel and interrupt.



**Figure 11-3. Functional operation of TZASC**

### NOTE

The *CoreLink TrustZone Address Space Controller (TZC-380) Supplement to AMBA Designer (ADR-301) User Guide* provides information about how to configure the controller.

### 11.4.1.1 Regions

A region is a contiguous area of address space. The TZASC provides each region with a programmable security permissions field. The security permissions value is used to enable the TZASC to either accept or deny a transaction access to that region. The transaction's secure vs non-secure attributes are used to determine the security settings of that transaction.

The TZASC always provides two regions, region 0 and region 1, and you can configure it to provide additional regions. With the exception of region 0, the TZASC enables you to program the following operating parameters for each region:

- Region enable.
- Security permissions.
- Base address.
- Size. The minimum address size of a region is 32KB.
- Subregion disable. See [Subregions](#).

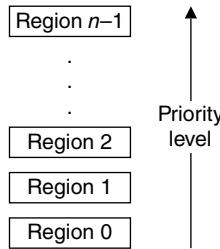
**NOTE**

Region 0 is known as the background region because it occupies the total memory space. You can program the security permissions of region 0, but the following parameters are fixed:

- **Base address:**0x0
- **Size:**The AXI\_ADDRESS\_MSB configuration parameter controls the address range of the TZASC, and therefore the region size.
- **Subregion disable:**This feature is not available for region 0.

### 11.4.1.2 Priority

The priority of a region is fixed and is determined by the region number. [Figure 11-4](#) shows how the priority of a region increases with the region number.

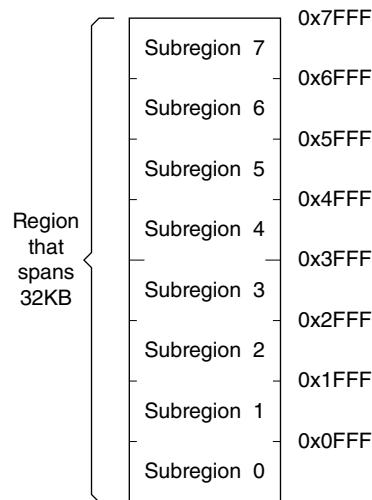


**Figure 11-4. Region priority**

When a transaction is received, its address is checked for a match with all the configured regions in turn. The order in which the regions are checked is determined by the priority level, the highest priority level is first. The first region that matches the transaction address match is used as the matching region. The matching regions security permission determines whether the transaction is permitted.

### 11.4.1.3 Subregions

The TZASC divides each region into eight equal-sized, non-overlapping subregions. [Figure 11-5](#) shows the subregions for an example region that is programmed to occupy an address span of 32KB.

**Figure 11-5. Subregion example**

#### 11.4.1.4 Subregion disable

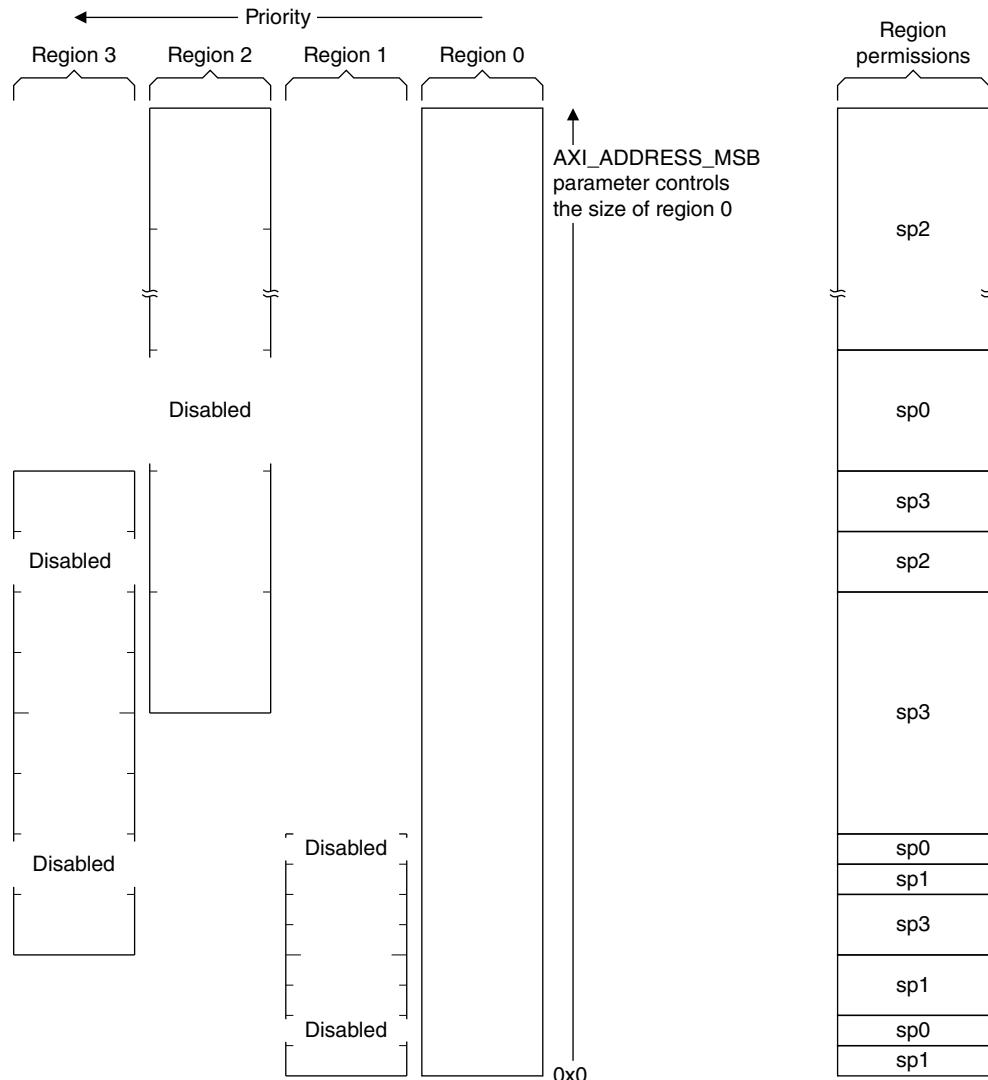
With the exception of region 0, you can program the TZASC to disable any or all of the eight subregions that comprise a region. When a subregion is disabled, the security permissions for its address range are provided by the next highest priority region that overlaps the address range.

##### Example configuration for subregion disable

[Figure 11-6](#) shows an example configuration that supports four regions, where:

- region 2 and region 3 are partially overlapped
- region 1 and region 3 are partially overlapped
- region 0 is overlapped with all regions.

With some subregions of region 1, region 2, and region 3 are disabled, and the resulting region permissions of the entire address space is shown in the [Figure 11-6](#).

**Figure 11-6. Subregion disable example****NOTE**

In [Figure 11-6](#)

- all subregions are enabled unless otherwise stated
- $sp_n$  represents the region permissions of region  $n$ .

### 11.4.1.5 Region security permissions

The TZASC enables you to program the security access permissions for any region that it is configured. A region is assigned a security permissions field,  $sp_{<n>}$ , in its `region_attributes_<n>` Register that enables you to have complete control of the permissions for that region. See [Register Descriptions](#).

## Security inversion

There are two modes of operation for the region security permissions, with or without security inversion.

By default, if you program a region to support non-secure accesses, the TZASC ensures that region must also support secure accesses. For example, if you program the region permissions for region 3 to be non-secure read only, the TZASC permits access to region 3 for secure reads and non-secure reads.

If you require that some regions are not accessible to masters in Secure state, but are accessible in Non-secure state, then you must enable security inversion.

See [Region security permissions](#) and [Security Inversion Register \(security\\_inversion\\_en\)](#) for more information.

### Programming security permissions when security inversion is disabled

By default, security inversion is disabled and therefore the TZASC only permits you to program certain combinations of security permissions. These combinations ensure that a master in Secure state is not denied access to a region that is programmed to only accept non-secure accesses. [Table 11-16](#) shows the possible security permissions when security inversion is disabled.

**Table 11-16. Region security permissions when security inversion is disabled**

	sp<n> field controls if the TZASC permits access for the following AXI transactions			
sp<n> field <sup>1</sup>	Secure read	Secure write	Non-secure read	Non-secure write
b0000	No	No	No	No
b0100	No	Yes	No	No
b0001, b0101	No	Yes	No	Yes
b1000	Yes	No	No	No
b0010, b1010	Yes	No	Yes	No
b1100	Yes	Yes	No	No
b1001, b1101	Yes	Yes	No	Yes
b0110, b1110	Yes	Yes	Yes	No
b0011, b0111, b1011, b1111	Yes	Yes	Yes	Yes

1. See [Region Attributes 0 Register \(region\\_attributes\\_0\)](#) for programming information.

### Programming security permissions when security inversion is enabled

## Functional description

If you enable security inversion, the TZASC permits you to program any combination of security permissions as [Table 11-17](#) shows.

**Table 11-17. Region security permissions when security inversion is enabled**

		sp<n> field controls if the TZASC permits access for the following AXI transactions			
sp<n> field <sup>1</sup>		Secure read	Secure write	Non-secure read	Non-secure write
b0000		No	No	No	No
b0001		No	No	No	Yes
b0010		No	No	Yes	No
b0011		No	No	Yes	Yes
b0100		No	Yes	No	No
b0101		No	Yes	No	Yes
b0110		No	Yes	Yes	No
b0111		No	Yes	Yes	Yes
b1000		Yes	No	No	No
b1001		Yes	No	No	Yes
b1010		Yes	No	Yes	No
b1011		Yes	No	Yes	Yes
b1100		Yes	Yes	No	No
b1101		Yes	Yes	No	Yes
b1110		Yes	Yes	Yes	No
b1111		Yes	Yes	Yes	Yes

1. See [Region Attributes 0 Register \(region\\_attributes\\_0\)](#) for programming information.

[Table 11-18](#) shows a typical example of memory map along with the register programming. The TZASC is configured to have 16 regions.

**Table 11-18. Typical example of memory map along with the register programming**

Region	Region <sup>1</sup>	Lock	Starting address	Region size	Region size	Sp <sup>2</sup>	Description
Region_0 (Default)	Enable	No	0x0	max	-	1100	Secure Read Write access (RW).
Region_1	Enable	No	0x0	64MB	b011001	1111	Non-secure Read or Write access (R/W), Secure R/W.

Table continues on the next page...

**Table 11-18. Typical example of memory map along with the register programming (continued)**

Region_2	Enable	No	0x0	16MB	b010111	1110	Non-secure <i>Read Only</i> access (RO), Secure RW for the normal world OS kernel.
Region_3	Enable	No	0x3D00000	512KB	b010010	1111	Regularly switched Non- secure, or Secure RW for a more complex shared memory buffers.
Region_4	Enable	No	0x3D80000	512KB	b010010	1100	Non-secure <i>No Access</i> (NA), Secure RW, a dedicated area for secure LCD Controller frame buffer.
Region_5	Enable	No	0x80000000	32KB	b001110	1111	Non-secure RW, Secure RW for address range of general peripherals such as screen control, and keyboard hardware.
Region_6	Enable	Yes	0x3C00000	512KB	b010010	1011	Non-secure RW, Secure RO for streaming from the normal world to the secure world.
Region_7	Enable	Yes	0x3C80000	512KB	b010010	1110	Non-secure RO, Secure RW for streaming from the

Table continues on the next page...

## Functional description

**Table 11-18. Typical example of memory map along with the register programming (continued)**

							secure world to the normal world.
Region_8	Enable	Yes	0x3E00000	512KB	b010010	1000	Non-secure NA, Secure RO for the secure world OS kernel.
Region_9	Enable	Yes	0x3E80000	512KB	b010010	1100	Non-secure NA, Secure RW for the secure world OS applications, heap, and stacks.
Region_10	Enable	Yes	0x3F00000	1MB	b010011	1100	Non-secure NA, Secure RW for Secure world OS applications, heap, and stacks.
Region_11 3	Enable	Yes	0x80008000	32KB	b001110	1100	Non-secure NA, Secure RW for address range of secure peripherals such as <i>Random Number Generator</i> (RNG), and cryptography support hardware.
Region_12	Enable	Yes	0xF0000000	256MB	b011011	0011	Non-secure RW, Secure NA for FLASH holding normal world OS plus disk.
Region_13	Enable	Yes	0xF0000000	1MB	b010011	1100	Non-secure NA, Secure RW for FLASH for secure boot, secure world

Table continues on the next page...

**Table 11-18. Typical example of memory map along with the register programming (continued)**

							OS, secure configuration details.
Region_14	Disable	-	-	-	-	-	-
Region_15	Disable	-	-	-	-	-	-

1. Region can be either Enable or Disable.
2. Security Permission (sp).
3. In a more typical system, these devices would be protected by a TrustZone Protection Controller (BP147), and associated TrustZone aware AXI to APB Bridges (BP135).

#### NOTE

The implementers system design, and security requirements are taken into account for this example. And any actual software programming must depend on the system where TZASC is plugged.

#### 11.4.1.6 Denied AXI transactions

If an AXI transaction has insufficient security privileges then for:

- **Reads:** The TZASC responds to the master by setting all bits of the read data bus to zero.

#### NOTE

If the TZASC is programmed to perform speculative accesses, it discards the data that it receives on read data.

- **Writes:** The TZASC prevents the transfer of data from the master to the slave by discarding the data of write data bus. If you program the TZASC to perform speculative accesses, it modifies the transfer to the slave by setting all bits of the:
  - write data bus to zero
  - write data strobe to zero.

#### NOTE

The action Register controls whether the TZASC signals to the master when a region permission failure occurs, and if so, the type of response it provides. See [Action register \(action\)](#).

### 11.4.1.7 Speculative accesses

By default, the TZASC performs read or write speculative accesses that means it forwards an AXI transaction address to a slave, before it verifies that the AXI transaction is permitted to read address or write address respectively.

The TZASC only permits the transfer of data between its AXI bus interfaces, after verifying the access that the read or write access is permitted respectively. If the verification fails, then it prevents the transfer of data between the master and slave as [Denied AXI transactions](#) describes.

You can disable speculative accesses by programming the speculation\_control Register. See [Speculation Control Register \(speculation\\_control\)](#). When speculative accesses are disabled, the TZASC verifies the permissions of the access before it forwards the access to the slave. If the TZASC:

- Permits the access, it commences an AXI transaction to the slave, and it adds one clock latency.
- Denies the access, it prevents the transfer of data between the master and slave as [Denied AXI transactions](#) describes. In this situation, the slave is unaware when the TZASC prevents the master from accessing the slave.

#### NOTE

Enabling speculative access is a potential security risk, if the device that is being protected reacts to this transaction. Most devices do not have to react to this level of access, and speculative access is much faster than validating the address before issuing the transaction.

### 11.4.1.8 Preventing writes to registers and using secure\_boot\_lock

By suitably programming lockdown Register, see [Lockdown Select Register \(lockdown\\_select\)](#), and asserting secure\_boot\_lock signal makes the following registers read only:

- speculation\_control Register. See [Speculation Control Register \(speculation\\_control\)](#).
- security\_inversion\_en Register. See [Security Inversion Register \(security\\_inversion\\_en\)](#).
- lockdown\_range Register. See [Lockdown Range Register \(lockdown\\_range\)](#).

#### Locking down the region using lockdown\_range and lockdown\_select registers

By programming the lockdown\_select, and lockdown\_range registers, and asserting the secure\_boot\_lock signal, you can lockdown the behavior of the TZASC so that it prevents unintentional or erroneous write to the regions specified in the lockdown\_range Register. However, read access to those regions is permitted:

- region\_setup\_low\_<n> Register. See [Region Setup Low 0 Register \(region\\_setup\\_low\\_0\)](#)
- region\_setup\_high\_<n> Register. See [Region Setup High 0 Register \(region\\_setup\\_high\\_0\)](#)
- region\_attributes\_<n> Register. See [Region Attributes 0 Register \(region\\_attributes\\_0\)](#).

The TZASC expects the secure\_boot\_lock signal to be asserted for at least one clock cycle. One clock after the secure\_boot\_lock is sampled HIGH by TZASC, then the registers mentioned in *Locking down the region using lockdown\_range and lockdown\_select registers* cannot be written, unless the TZASC is reset by asserting aresetn.

### 11.4.1.9 Using locked transaction sequences

If a master performs locked transaction sequences, a transaction might stall, or an AXI protocol violation might occur when:

#### Transaction sequence crosses a 4 KB boundary

If a locked transaction sequence crosses a 4 KB boundary and the regions have different region permissions, the TZASC might prevent access to the second region and therefore the slave would not receive the latter part of the locked transaction sequence.

#### NOTE

The AXI protocol recommends that locked transaction sequences do not cross a 4 KB address boundary.

#### Secure state change

During a locked transaction sequence, if a master changes the state of secure and non-secure attributes and the region has different region permissions for Secure state and Non-secure state, the TZASC might deny a transaction and therefore the slave would not receive the latter part of the locked transaction sequence.

#### Reads and writes

During a locked transaction sequence, if a master performs reads and writes to a region, depending on the region permissions, the TZASC might deny a transaction and therefore the slave would not receive the latter part of the locked transaction sequence.

#### **11.4.1.10 Using exclusive accesses**

If a master performs exclusive accesses to an address region, you must program the TZASC to permit read and write accesses to that address region, for the expected settings of secure and non-secure attributes, otherwise the read or write transaction might fail.

### **11.4.2 Constraints of use**

The TZASC has the following considerations relating to change in programmers view on an active system:

- When changing the setting of a TZASC region,
  - The current accepted AXI transaction, if it falls into that region, would act according to the previous settings for that region.
  - Any other outstanding AXI transactions, that falls into that region, would effect by the new settings for that region.
- Given little ability to predict that the mentioned AXI transactions would effect, it is obviously desirable that there are no outstanding AXI transactions when a regions setting are changed.
  - In simple systems this can potentially be achieved by the core not accessing the given region during the period of the cores transition between security states. Even in these cases, the status of cached data and instructions needs to be considered.
  - In more complicated systems the code that changes the TZASC region settings must have to inform other AXI bus masters to desist or complete acting on that region before performing the region setting changes. After having such an action acknowledged the code must also have to instigate a suitable delay before then acting.

An example of this can be an LCD controller dealing with a frame buffer that is switching between a Normal world and Secure world use.

#### **NOTE**

There is no direct mechanism to ascertain if there are any outstanding AXI transactions, and so the designer must use their system knowledge to apply reasonable mechanisms.

It is recommended that any DECERR, or TZASC interrupt handler is designed to expect, and potentially ignore events generated under these circumstances.



# Chapter 12

## Supplemental Configuration Unit

### 12.1 Introduction

The supplemental configuration unit provides device specific configuration and status registers for the device. It is the chip defined module for extending the device configuration unit (DCFG) module. It provides a set of CCSR space registers in addition to those available in the device configuration unit. There are no source and target IDs associated with this unit

### 12.2 Overview

The supplement configuration unit contains the following registers:

- Chip specific control and status registers (CCSR)
- Pinmux control registers(CCSR)

### 12.3 SCFG Memory Map/Register Definition

SCFG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
157_0070	USB1 Parameter 1 Control Register (SCFG_USB1PRM1CR)	32	R/W	25E7_2B2Ah	<a href="#">12.3.1/412</a>
157_0074	USB1 Parameter 2 Control Register (SCFG_USB1PRM2CR)	32	R/W	17C1_FF48h	<a href="#">12.3.2/414</a>
157_0078	USB1 Parameter 3 Control Register (SCFG_USB1PRM3CR)	32	R/W	0000_0000h	<a href="#">12.3.3/415</a>
157_007C	USB2 Parameter 1 Control Register (SCFG_USB2PRM1CR)	32	R/W	25E7_2B2Ah	<a href="#">12.3.4/416</a>

Table continues on the next page...

## SCFG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
157_0080	USB2 Parameter 2 Control Register (SCFG_USB2PRM2CR)	32	R/W	17C1_FF48h	<a href="#">12.3.5/418</a>
157_0084	USB2 Parameter3 Control Register (SCFG_USB2PRM3CR)	32	R/W	0000_0000h	<a href="#">12.3.6/419</a>
157_0088	USB3 Parameter 1 Control Register (SCFG_USB3PRM1CR)	32	R/W	25E7_2B2Ah	<a href="#">12.3.7/420</a>
157_008C	USB3 Parameter 2 Control Register (SCFG_USB3PRM2CR)	32	R/W	17C1_FF48h	<a href="#">12.3.8/422</a>
157_0090	USB3 Parameter 3 Control Register (SCFG_USB3PRM3CR)	32	R/W	0000_0000h	<a href="#">12.3.9/423</a>
157_0100	USB2 ICID Register (SCFG_USB2_ICID)	32	R/W	0000_0000h	<a href="#">12.3.10/424</a>
157_0104	USB3 ICID Register (SCFG_USB3_ICID)	32	R/W	0000_0000h	<a href="#">12.3.11/425</a>
157_0114	qDMA ICID Register (SCFG_DMA_ICID)	32	R/W	0000_0000h	<a href="#">12.3.12/426</a>
157_0118	SATA ICID Register (SCFG_SATA_ICID)	32	R/W	0000_0000h	<a href="#">12.3.13/427</a>
157_011C	USB1 ICID Register (SCFG_USB1_ICID)	32	R/W	0000_0000h	<a href="#">12.3.14/428</a>
157_0120	QE ICID Register (SCFG_QE_ICID)	32	R/W	0000_0000h	<a href="#">12.3.15/429</a>
157_0124	eSDHC ICID Register (SCFG_SDHC_ICID)	32	R/W	0000_0000h	<a href="#">12.3.16/430</a>
157_0128	eDMA ICID Register (SCFG_eDMA_ICID)	32	R/W	0000_0000h	<a href="#">12.3.17/431</a>
157_012C	ETR ICID Register (SCFG_ETR_ICID)	32	R/W	0000_0000h	<a href="#">12.3.18/432</a>
157_0130	Core 0 soft reset Register (SCFG_CORE0_SFT_RST)	32	R/W	0000_0000h	<a href="#">12.3.19/433</a>
157_0134	Core 1 soft reset Register (SCFG_CORE1_SFT_RST)	32	R/W	0000_0000h	<a href="#">12.3.20/434</a>
157_0138	Core 2 soft reset Register (SCFG_CORE2_SFT_RST)	32	R/W	0000_0000h	<a href="#">12.3.21/435</a>
157_013C	Core 3soft reset Register (SCFG_CORE3_SFT_RST)	32	R/W	0000_0000h	<a href="#">12.3.22/436</a>
157_0144	PEX PME control register (SCFG_PEXPMECR)	32	R/W	0000_0000h	<a href="#">12.3.23/437</a>
157_0154	FTM chain configuration (SCFG_FTM_CHAIN_CONFIG)	32	R/W	0000_0000h	<a href="#">12.3.24/438</a>
157_0158	ALTCBAR Register (SCFG_ALTCBAR)	32	R/W	0000_0000h	<a href="#">12.3.25/439</a>
157_015C	QSPI CONFIG Register (SCFG_QSPI_CFG)	32	R/W	0010_0000h	<a href="#">12.3.26/439</a>
157_016C	QOS1 Register (SCFG_QOS1)	32	R/W	0000_0000h	<a href="#">12.3.27/440</a>
157_0170	QOS2 Register (SCFG_QOS2)	32	R/W	0000_0000h	<a href="#">12.3.28/441</a>
157_0188	GIC-400 Address 64K Page Alignment Register (SCFG_GIC400_ADDR_ALIGN_64K)	32	R/W	0700_0000h	<a href="#">12.3.29/442</a>
157_018C	Debug ICID Register (SCFG_Debug_ICID)	32	R/W	0700_0000h	<a href="#">12.3.30/443</a>
157_01A4	Snoop Configuration Register (SCFG_SNPCNFGCR)	32	R/W	0000_0000h	<a href="#">12.3.31/444</a>
157_01AC	Interrupt Polarity Register (SCFG_INTPCR)	32	R/W	0000_0000h	<a href="#">12.3.32/446</a>
157_0204	CORE Soft Reset Enable Register (SCFG_CORESRENCR)	32	R/W	0000_0000h	<a href="#">12.3.33/448</a>
157_0220	Core 0 Reset Vector Base Address0 (SCFG_RVBAR0_0)	32	R/W	0000_0000h	<a href="#">12.3.34/448</a>
157_0224	Core 0 Reset Vector Base Address1 (SCFG_RVBAR0_1)	32	R/W	0000_0000h	<a href="#">12.3.35/449</a>
157_0228	Core 1 Reset Vector Base Address0 (SCFG_RVBAR1_0)	32	R/W	0000_0000h	<a href="#">12.3.36/450</a>
157_022C	Core 1 Reset Vector Base Address1 (SCFG_RVBAR1_1)	32	R/W	0000_0000h	<a href="#">12.3.37/450</a>
157_0230	Core 2 Reset Vector Base Address0 (SCFG_RVBAR2_0)	32	R/W	0000_0000h	<a href="#">12.3.38/451</a>

Table continues on the next page...

**SCFG memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
157_0234	Core 2 Reset Vector Base Address1 (SCFG_RVBAR2_1)	32	R/W	0000_0000h	<a href="#">12.3.39/451</a>
157_0238	Core 3 Reset Vector Base Address0 (SCFG_RVBAR3_0)	32	R/W	0000_0000h	<a href="#">12.3.40/452</a>
157_023C	Core 3 Reset Vector Base Address1 (SCFG_RVBAR3_1)	32	R/W	0000_0000h	<a href="#">12.3.41/452</a>
157_0240	Core Low Power Mode Control Status Register (SCFG_LPMCSR)	32	R/W	0101_0101h	<a href="#">12.3.42/453</a>
157_0404	ECGTX Clock Mux Control Register (SCFG_ECGTXCMCR)	32	R/W	0000_0000h	<a href="#">12.3.43/455</a>
157_0408	SDHC IO VSEL Control Register (SCFG_SDHCIOWSELCR)	32	R/W	0000_0000h	<a href="#">12.3.44/456</a>
157_040C	Extended RCW PinMux Control Register (SCFG_RCWPMUXCR0)	32	R/W	0000_0000h	<a href="#">12.3.45/457</a>
157_0410	USB DRVVBUS Control Register (SCFG_USBDRVVBUS_SELCR)	32	R/W	0000_0000h	<a href="#">12.3.46/459</a>
157_0414	USB PWRFAULTControl Register (SCFG_USBPWRFAULT_SELCR)	32	R/W	0000_0000h	<a href="#">12.3.47/459</a>
157_0418	USB PHY1 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR1)	32	R/W	8000_009Eh	<a href="#">12.3.48/460</a>
157_041C	USB PHY2 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR2)	32	R/W	8000_009Eh	<a href="#">12.3.49/461</a>
157_0420	USB PHY3 Reference Clock Select Register (SCFG_USB_REFCLK_SELCR3)	32	R/W	8000_009Eh	<a href="#">12.3.50/462</a>
157_0424	Retention Request Control Register (SCFG_RETREQCR)	32	R/W	0000_0000h	<a href="#">12.3.51/463</a>
157_042C	CORE PM Control Register (SCFG_COREPMCR)	32	R/W	0000_0000h	<a href="#">12.3.52/464</a>
157_0600	SCRATCHRWn - Scratch Read Write Registers (SCFG_SCRATCHRW0)	32	R/W	0000_0000h	<a href="#">12.3.53/464</a>
157_0604	SCRATCHRWn - Scratch Read Write Registers (SCFG_SCRATCHRW1)	32	R/W	0000_0000h	<a href="#">12.3.53/464</a>
157_0608	SCRATCHRWn - Scratch Read Write Registers (SCFG_SCRATCHRW2)	32	R/W	0000_0000h	<a href="#">12.3.53/464</a>
157_060C	SCRATCHRWn - Scratch Read Write Registers (SCFG_SCRATCHRW3)	32	R/W	0000_0000h	<a href="#">12.3.53/464</a>
157_0680	Core Boot Control Register (SCFG_COREBCR)	32	R/W	FFFF_FFFFh	<a href="#">12.3.54/465</a>
157_1000	Shared Message Signaled Interrupt Index Register (SCFG_G0MSIIR)	32	R/W	0000_0000h	<a href="#">12.3.55/466</a>
157_1010	Shared Message Signaled Interrupt Register (SCFG_G0MSIR1)	32	R	0000_0000h	<a href="#">12.3.56/468</a>
157_1014	Shared Message Signaled Interrupt Register (SCFG_G0MSIR2)	32	R	0000_0000h	<a href="#">12.3.57/468</a>
157_1018	Shared Message Signaled Interrupt Register (SCFG_G0MSIR3)	32	R	0000_0000h	<a href="#">12.3.58/469</a>
157_101C	Shared Message Signaled Interrupt Register (SCFG_G0MSIR4)	32	R	0000_0000h	<a href="#">12.3.59/469</a>
157_2000	Shared Message Signaled Interrupt Index Register (SCFG_G1MSIIR)	32	R/W	0000_0000h	<a href="#">12.3.60/470</a>

Table continues on the next page...

## SCFG memory map (continued)

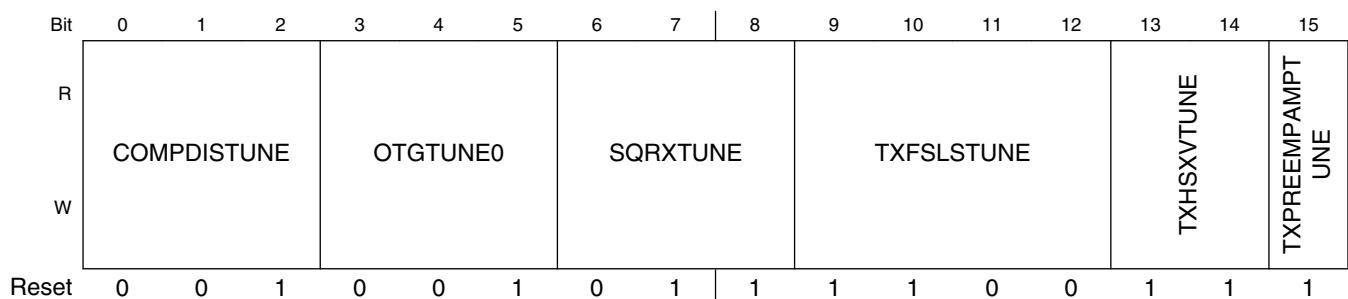
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
157_2010	Shared Message Signaled Interrupt Register (SCFG_G1MSIR1)	32	R	0000_0000h	<a href="#">12.3.61/472</a>
157_2014	Shared Message Signaled Interrupt Register (SCFG_G1MSIR2)	32	R	0000_0000h	<a href="#">12.3.62/472</a>
157_2018	Shared Message Signaled Interrupt Register (SCFG_G1MSIR3)	32	R	0000_0000h	<a href="#">12.3.63/473</a>
157_201C	Shared Message Signaled Interrupt Register (SCFG_G1MSIR4)	32	R	0000_0000h	<a href="#">12.3.64/473</a>
157_3000	Shared Message Signaled Interrupt Index Register (SCFG_G2MSIIR)	32	R/W	0000_0000h	<a href="#">12.3.65/474</a>
157_3010	Shared Message Signaled Interrupt Register (SCFG_G2MSIR1)	32	R	0000_0000h	<a href="#">12.3.66/476</a>
157_3014	Shared Message Signaled Interrupt Register (SCFG_G2MSIR2)	32	R	0000_0000h	<a href="#">12.3.67/476</a>
157_3018	Shared Message Signaled Interrupt Register (SCFG_G2MSIR3)	32	R	0000_0000h	<a href="#">12.3.68/477</a>
157_301C	Shared Message Signaled Interrupt Register (SCFG_G2MSIR4)	32	R	0000_0000h	<a href="#">12.3.69/477</a>

### 12.3.1 USB1 Parameter 1 Control Register (SCFG\_USB1PRM1CR)

This register contains the USB1 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB1 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB1 interface.

Address: 157\_0000h base + 70h offset = 157\_0070h



Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TXPREEMPAMPTUNE	TXPREEMPPULSE		TXRESTTUNE		TXRESETUNE		TXVREFTUNE						PCSTXDEEMPH3P5DB		
W																
Reset	0	0	1	0	1	0	1	1	0	0	1	0	1	0	1	0

**SCFG\_USB1PRM1CR field descriptions**

Field	Description
0–2 COMPDISTUNE	<p>It drives the value of USB3 COMPDISTUNE signal.</p> <p>Disconnect threshold adjustment: It adjusts the voltage level for the threshold used to detect a disconnect event at the host.</p>
3–5 OTGTUNE0	<p>It drives the value of USB3 OTGTUNE0 signal.</p> <p>V<sub>BUS</sub> valid threshold adjustment: This bus adjusts the voltage level for the V<sub>BUS</sub> valid threshold.</p>
6–8 SQRXTUNE	<p>It drives the value of USB3 SQRXTUNE signal.</p> <p>Squelch threshold adjustment: It adjusts the voltage level for the threshold used to detect valid high speed data.</p>
9–12 TXFSLSTUNE	<p>It drives the value of USB3 TXFSLSTUNE signal.</p> <p>FS/LS source impedance adjustment: It adjusts the low and full speed single-ended source impedance while driving high.</p>
13–14 TXHSXVTUNE	<p>It drives the value of USB3 TXHSXVTUNE signal.</p> <p>Transmitter high speed crossover adjustment: This bus adjusts the voltage at which the D+ and D- signals cross while transmitting in HS mode.</p> <p>11 Default</p> <p>10 + 15 mV</p> <p>01 – 15 mV</p> <p>00 Reserved</p>
15–16 TXPREEMPAMPTUNE	<p>It drives the value of USB3 TXPREEMPAMPTUNE signal.</p> <p>HS transmitter pre-emphasis current control: This signal controls the amount of current sourced to D+ and D- after a J-to-K or K-to-J transition. The HS transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600 µA and is defined as 1X pre-emphasis current.</p> <p>11 HS transmitter pre-emphasis circuit sources 3X pre-emphasis current</p> <p>10 HS transmitter pre-emphasis circuit sources 2X pre-emphasis current</p> <p>01 HS transmitter pre-emphasis circuit sources 1X preemphasis current</p> <p>00 HS transmitter pre-emphasis disabled (Default)</p>
17 TXPREEMPPULSETUNE	<p>It drives the value of USB3 TXPREEMPPULSETUNE signal.</p> <p>HS transmitter pre-emphasis duration control: This signal controls the duration for which the HS pre-emphasis current is sourced onto D+ or D-. The HS transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580 ps and is defined as 1X pre-emphasis duration. This signal is valid only if either TXPREEMPAMPTUNE[1] or TXPREEMPAMPTUNE[0] is set to 1'b1.</p>

*Table continues on the next page...*

**SCFG\_USB1PRM1CR field descriptions (continued)**

Field	Description
	1 1X, short pre-emphasis current duration 0 2X (Default), long pre-emphasis current duration
18–19 TXRESTUNE	It drives the value of USB3 TXRESTUNE signal. USB source impedance adjustment: This bus adjusts the driver source impedance to compensate for added series resistance on the USB. 11 Source impedance is decreased by approximately 4 Ω. 10 Source impedance is decreased by approximately 2 Ω. 01 Default 00 Source impedance is increased by approximately 1.5 Ω.
20–21 TXRISETUNE	It drives the value of USB3 TXRISETUNE signal. HS transmitter rise/fall time adjustment: It adjusts the rise/fall times of the high-speed waveform.
22–25 TXVREFTUNE	It drives the value of USB3 TXVREFTUNE signal. HS DC voltage level adjustment: It adjusts the high speed DC level voltage.
26–31 PCSTXDEEMPH3P5DB	It drives the value of USB3 pcs_tx_deemph_3p5db (Tx de-emphasis at 3.5 dB) signal.

**12.3.2 USB1 Parameter 2 Control Register (SCFG\_USB1PRM2CR)**

The USB1 parameter 2 control register contains the USB1 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB1 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB1 interface.

Address: 157\_0000h base + 74h offset = 157\_0074h

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R W	PCSRXLOSMASKVAL														PCSTXDEEMPH6DB		
Reset	0	0	0	1	0	1	1	1		1	1	0	0	0	0	0	1
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R W	PCSTXSWINGFULL							LOSBIAS			TXVBOOSTLVL			—			
Reset	1	1	1	1	1	1	1	1		0	1	0	0	1	0	0	0

**SCFG\_USB1PRM2CR field descriptions**

Field	Description
0–9 PCSRXLOSMASKVAL	It drives the value of USB3 pcs_rx_los_mask_val signal. Configurable loss-of-signal mask width: It sets the number of reference clock cycles to mask the incoming LFPS in U3 and U2 states. It masks the incoming LFPS for the number of reference clock cycles equal to the value of pcs_rx_los_mask_val[9:0]. This control filters out short, non-compliant LFPS glitches sent by a non-compliant host.

*Table continues on the next page...*

**SCFG\_USB1PRM2CR field descriptions (continued)**

Field	Description
	If this bus is set to 10'b0, it disables masking. This bus should be accessible to general configuration registers for system testing and debug. The value should be defined only in reset. Changing this value during operation might disrupt normal operation of the link.
10–15 PCSTXDEEMPH6DB	It drives the value of USB3 pcs_tx_deemph_6db signal. Tx de-emphasis at 6 dB: This bus is provided for completeness and as a second potential launch amplitude.
16–22 PCSTXSWINGFULL	It drives the value of USB3 pcs_tx_swing_full signal. Tx amplitude (full swing mode): This static value sets the launch amplitude of the transmitter.
23–25 LOSBIAS	It drives the value of USB3 los_bias signal. Loss-of-signal detector threshold level control: It sets the LOS detection threshold level. A positive binary bit setting change results in a +15 mVp incremental change in the LOS threshold. A negative binary bit setting change results in a -15 mVp incremental change in the LOS threshold. The 3b'000 setting is reserved and must not be used.
26–28 TXVBOOSTLVL	It drives the value of USB3 tx_vboost_lvl signal. Tx voltage boost level: It sets the boosted transmit launch amplitude (mVppd).
29–31 —	Reserved

**12.3.3 USB1 Parameter 3 Control Register (SCFG\_USB1PRM3CR)**

The USB1 parameter 3 control register contains the USB1 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB1 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB1 interface.

Address: 157\_0000h base + 78h offset = 157\_0078h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_USB1PRM3CR field descriptions**

Field	Description
0–4 USB1ACJT	Drives the value of USB3 acjt_level signal
5–6 VATESTENB	Drives the value of USB3 PHY VATESTENB signal
7 LPBKENB0	Drives the value of USB3 PHY LOOPBACKENB0 signal
8 LNTX2RXLPBK	Drives the value of all 3 USB3 PHYs lane0_tx2rx_lopbk signal
9–15 mPLL_MULT	Drives the value of USB3 mpll_multiplier signal
16–31 —	Reserved

**12.3.4 USB2 Parameter 1 Control Register (SCFG\_USB2PRM1CR)**

The USB2 parameter 1 control register contains the USB2 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB2 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB2 interface.

Address: 157\_0000h base + 7Ch offset = 157\_007Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	COMPDISTUNE			OTGTUNE0			SQRXTUNE			TXFSLSTUNE			TXHSXVTUNE		TXPREEMPAMPTUNE		
W																	
Reset	0	0	1	0	0	1	0	1	1	1	1	0	0	1	1	1	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	TXPREEMPAMPTUNE	TXPREMPULSE	TRESTUNE	TXRSETUNE		TXVREFTUNE			PCSTXDEEMPH3P5DB								
W																	
Reset	0	0	1	0	1	0	1	1	0	0	1	0	1	0	1	0	

**SCFG\_USB2PRM1CR field descriptions**

Field	Description
0–2 COMPDISTUNE	<p>It drives the value of USB3 COMPDISTUNE signal.</p> <p>Disconnect threshold adjustment: It adjusts the voltage level for the threshold used to detect a disconnect event at the host.</p>
3–5 OTGTUNE0	<p>It drives the value of USB3 OTGTUNE0 signal.</p> <p><math>V_{BUS}</math> valid threshold adjustment: This bus adjusts the voltage level for the <math>V_{BUS}</math> valid threshold.</p>
6–8 SQRXTUNE	<p>It drives the value of USB3 SQRXTUNE signal.</p> <p>Squelch threshold adjustment: It adjusts the voltage level for the threshold used to detect valid high speed data.</p>
9–12 TXFSLSTUNE	<p>It drives the value of USB3 TXFSLSTUNE signal.</p> <p>FS/LS source impedance adjustment: It adjusts the low and full speed single-ended source impedance while driving high.</p>
13–14 TXHSXVTUNE	<p>It drives the value of USB3 TXHSXVTUNE signal.</p> <p>Transmitter high speed crossover adjustment: This bus adjusts the voltage at which the D+ and D- signals cross while transmitting in HS mode.</p> <p>11 Default 10 + 15 mV 01 – 15 mV 00 Reserved</p>
15–16 TXPREEMPAMPTUNE	<p>It drives the value of USB3 TXPREEMPAMPTUNE signal.</p> <p>HS transmitter pre-emphasis current control: This signal controls the amount of current sourced to D+ and D- after a J-to-K or K-to-J transition. The HS transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600 <math>\mu</math>A and is defined as 1X pre-emphasis current.</p> <p>11 HS transmitter pre-emphasis circuit sources 3X pre-emphasis current 10 HS transmitter pre-emphasis circuit sources 2X pre-emphasis current 01 HS transmitter pre-emphasis circuit sources 1X preemphasis current 00 HS transmitter pre-emphasis disabled (Default)</p>
17 TXPREEMPPULSETUNE	<p>It drives the value of USB3 TXPREEMPPULSETUNE signal.</p> <p>HS transmitter pre-emphasis duration control: This signal controls the duration for which the HS pre-emphasis current is sourced onto D+ or D-. The HS transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580 ps and is defined as 1X pre-emphasis duration. This signal is valid only if either TXPREEMPAMPTUNE[1] or TXPREEMPAMPTUNE[0] is set to 1'b1.</p> <p>1 1X, short pre-emphasis current duration 0 2X (Default), long pre-emphasis current duration</p>
18–19 TXRESTUNE	<p>It drives the value of USB3 TXRESTUNE signal.</p> <p>USB source impedance adjustment: This bus adjusts the driver source impedance to compensate for added series resistance on the USB.</p> <p>11 Source impedance is decreased by approximately 4 <math>\Omega</math>. 10 Source impedance is decreased by approximately 2 <math>\Omega</math>. 01 Default</p>

*Table continues on the next page...*

**SCFG\_USB2PRM1CR field descriptions (continued)**

Field	Description
	00 Source impedance is increased by approximately 1.5 Ω.
20–21 TXRISETUNE	It drives the value of USB3 TXRISETUNE signal. HS transmitter rise/fall time adjustment: It adjusts the rise/fall times of the high-speed waveform.
22–25 TXVREFTUNE	It drives the value of USB3 TXVREFTUNE signal. HS DC voltage level adjustment: It adjusts the high speed DC level voltage.
26–31 PCSTXDEEMPH3P5DB	It drives the value of USB3 pcs_tx_deemph_3p5db (Tx de-emphasis at 3.5 dB) signal.

**12.3.5 USB2 Parameter 2 Control Register (SCFG\_USB2PRM2CR)**

The USB2 parameter 2 control register contains the USB2 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB2 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB2 interface.

Address: 157\_0000h base + 80h offset = 157\_0080h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	PCSRXLOSMASKVAL										PCSTXDEEMPH6DB					
Reset	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	PCSTXSWINGFULL								LOSSBIAS		TXVBOOSTLVL			—		
Reset	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	0

**SCFG\_USB2PRM2CR field descriptions**

Field	Description
0–9 PCSRXLOSMASKVAL	It drives the value of USB3 pcs_rx_los_mask_val signal. Configurable loss-of-signal mask width: It sets the number of reference clock cycles to mask the incoming LFPS in U3 and U2 states. It masks the incoming LFPS for the number of reference clock cycles equal to the value of pcs_rx_los_mask_val[9:0]. This control filters out short, non-compliant LFPS glitches sent by a non-compliant host. If this bus is set to 10'b0, it disables masking. This bus should be accessible to general configuration registers for system testing and debug. The value should be defined only in reset. Changing this value during operation might disrupt normal operation of the link.
10–15 PCSTXDEEMPH6DB	It drives the value of USB3 pcs_tx_deemph_6db signal. Tx de-emphasis at 6 dB: This bus is provided for completeness and as a second potential launch amplitude.
16–22 PCSTXSWINGFULL	It drives the value of USB3 pcs_tx_swings_full signal. Tx amplitude (full swing mode): This static value sets the launch amplitude of the transmitter.

Table continues on the next page...

**SCFG\_USB2PRM2CR field descriptions (continued)**

Field	Description
23–25 LOSBIAS	It drives the value of USB3 los_bias signal. Loss-of-signal detector threshold level control: It sets the LOS detection threshold level. A positive binary bit setting change results in a +15 mVp incremental change in the LOS threshold. A negative binary bit setting change results in a -15 mVp incremental change in the LOS threshold. The 3b'000 setting is reserved and must not be used.
26–28 TXVBOOSTLVL	It drives the value of USB3 tx_vboost_lvl signal. Tx voltage boost level: It sets the boosted transmit launch amplitude (mVppd).
29–31 —	Reserved

**12.3.6 USB2 Parameter3 Control Register (SCFG\_USB2PRM3CR)**

The USB2 parameter 3 control register contains the USB2 parameters signals. This register is reset on HRESET

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB2 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB2 interface.

Address: 157\_0000h base + 84h offset = 157\_0084h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_USB2PRM3CR field descriptions**

Field	Description
0–4 USB2ACJT	Drives the value of USB2 acjt_level signal
5–6 VATESTENB	Drives the value of USB3 PHY VATESTENB signal
7 LPBKENB0	Drives the value of USB3 PHY LOOPBACKENB0 signal

Table continues on the next page...

**SCFG\_USB2PRM3CR field descriptions (continued)**

Field	Description
8 —	Reserved
9–15 mPLL_MULT	Drives the value of USB3 mpll_multiplier signal
16–31 —	Reserved

**12.3.7 USB3 Parameter 1 Control Register (SCFG\_USB3PRM1CR)**

The USB3 parameter 1 control register contains the USB3 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB3 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB3 interface.

Address: 157\_0000h base + 88h offset = 157\_0088h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
	COMPDISTUNE			OTGTUNE0			SQRXTUNE			TXFSLSTUNE						
W																
Reset	0	0	1	0	0	1	0	1	1	1	1	0	0	1	1	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
	TXPREEMPAMPTUNE	TXPREEMPPULSE		TXRESTUNE		TXRSETUNE		TXVREFTUNE			PCSTXDEEMPH3P5DB					
W																
Reset	0	0	1	0	1	0	1	1	0	0	1	0	1	0	1	0

**SCFG\_USB3PRM1CR field descriptions**

Field	Description
0–2 COMPDISTUNE	It drives the value of USB3 COMPDISTUNE signal. Disconnect threshold adjustment: It adjusts the voltage level for the threshold used to detect a disconnect event at the host.
3–5 OTGTUNE0	It drives the value of USB3 OTGTUNE0 signal.

*Table continues on the next page...*

**SCFG\_USB3PRM1CR field descriptions (continued)**

Field	Description
	$V_{BUS}$ valid threshold adjustment: This bus adjusts the voltage level for the $V_{BUS}$ valid threshold.
6–8 SQRXTUNE	It drives the value of USB3 SQRXTUNE signal. Squelch threshold adjustment: It adjusts the voltage level for the threshold used to detect valid high speed data.
9–12 TXFSLSTUNE	It drives the value of USB3 TXFSLSTUNE signal. FS/LS source impedance adjustment: It adjusts the low and full speed single-ended source impedance while driving high.
13–14 TXHSXVTUNE	It drives the value of USB3 TXHSXVTUNE signal. Transmitter high speed crossover adjustment: This bus adjusts the voltage at which the D+ and D- signals cross while transmitting in HS mode. 11 Default 10 + 15 mV 01 – 15 mV 00 Reserved
15–16 TXPREEMPAMPTUNE	It drives the value of USB3 TXPREEMPAMPTUNE signal. HS transmitter pre-emphasis current control: This signal controls the amount of current sourced to D+ and D- after a J-to-K or K-to-J transition. The HS transmitter pre-emphasis current is defined in terms of unit amounts. One unit amount is approximately 600 $\mu$ A and is defined as 1X pre-emphasis current. 11 HS transmitter pre-emphasis circuit sources 3X pre-emphasis current 10 HS transmitter pre-emphasis circuit sources 2X pre-emphasis current 01 HS transmitter pre-emphasis circuit sources 1X preemphasis current 00 HS transmitter pre-emphasis disabled (Default)
17 TXPREEMPPULSETUNE	It drives the value of USB3 TXPREEMPPULSETUNE signal. HS transmitter pre-emphasis duration control: This signal controls the duration for which the HS pre-emphasis current is sourced onto D+ or D-. The HS transmitter pre-emphasis duration is defined in terms of unit amounts. One unit of pre-emphasis duration is approximately 580 ps and is defined as 1X pre-emphasis duration. This signal is valid only if either TXPREEMPAMPTUNE[1] or TXPREEMPAMPTUNE[0] is set to 1'b1. 1 1X, short pre-emphasis current duration 0 2X (Default), long pre-emphasis current duration
18–19 TXRESTUNE	It drives the value of USB3 TXRESTUNE signal. USB source impedance adjustment: This bus adjusts the driver source impedance to compensate for added series resistance on the USB. 11 Source impedance is decreased by approximately 4 $\Omega$ . 10 Source impedance is decreased by approximately 2 $\Omega$ . 01 Default 00 Source impedance is increased by approximately 1.5 $\Omega$ .
20–21 TXRISETUNE	It drives the value of USB3 TXRISETUNE signal. HS transmitter rise/fall time adjustment: It adjusts the rise/fall times of the high-speed waveform.
22–25 TXVREFTUNE	It drives the value of USB3 TXVREFTUNE signal.

*Table continues on the next page...*

**SCFG\_USB3PRM1CR field descriptions (continued)**

Field	Description
	HS DC voltage level adjustment: It adjusts the high speed DC level voltage.
26–31 PCSTXDEEMPH3P5DB	It drives the value of USB3 pcs_tx_deemph_3p5db (Tx de-emphasis at 3.5 dB) signal.

**12.3.8 USB3 Parameter 2 Control Register (SCFG\_USB3PRM2CR)**

The USB3 parameter 2 control register contains the USB3 parameters signals. This register is reset on HRESET.

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB3 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB3 interface.

Address: 157\_0000h base + 8Ch offset = 157\_008Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	PCSRXLOSMASKVAL										PCSTXDEEMPH6DB					
Reset	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	PCSTXSWINGFULL								LOSBIAS		TXVBOOSTLVL			—		
Reset	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	0

**SCFG\_USB3PRM2CR field descriptions**

Field	Description
0–9 PCSRXLOSMASKVAL	It drives the value of USB3 pcs_rx_los_mask_val signal. Configurable loss-of-signal mask width: It sets the number of reference clock cycles to mask the incoming LFPS in U3 and U2 states. It masks the incoming LFPS for the number of reference clock cycles equal to the value of pcs_rx_los_mask_val[9:0]. This control filters out short, non-compliant LFPS glitches sent by a non-compliant host. If this bus is set to 10'b0, it disables masking. This bus should be accessible to general configuration registers for system testing and debug. The value should be defined only in reset. Changing this value during operation might disrupt normal operation of the link.
10–15 PCSTXDEEMPH6DB	It drives the value of USB3 pcs_tx_deemph_6db signal. Tx de-emphasis at 6 dB: This bus is provided for completeness and as a second potential launch amplitude.
16–22 PCSTXSWINGFULL	It drives the value of USB3 pcs_tx_swing_full signal. Tx amplitude (full swing mode): This static value sets the launch amplitude of the transmitter.
23–25 LOSBIAS	It drives the value of USB3 los_bias signal. Loss-of-signal detector threshold level control: It sets the LOS detection threshold level.

*Table continues on the next page...*

**SCFG\_USB3PRM2CR field descriptions (continued)**

Field	Description
	A positive binary bit setting change results in a +15 mVp incremental change in the LOS threshold. A negative binary bit setting change results in a -15 mVp incremental change in the LOS threshold. The 3b'000 setting is reserved and must not be used.
26–28 TXVBOOSTLVL	It drives the value of USB3 tx_vboost_lvl signal. Tx voltage boost level: It sets the boosted transmit launch amplitude (mVppd).
29–31 —	Reserved

**12.3.9 USB3 Parameter 3 Control Register (SCFG\_USB3PRM3CR)**

Note that the signals described in the register are described as big endian(0:x) however the signals in the USB3 interface are little endian(x:0). The connectivity is done in such a way that bit 0 of the following register field corresponds to bit 0 of the USB3 interface.

Address: 157\_0000h base + 90h offset = 157\_0090h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_USB3PRM3CR field descriptions**

Field	Description
0–4 USB3ACJT	Drives the value of USB3 acjt_level signal
5–6 VATESTENB	Drives the value of USB3 PHY VATESTENB signal
7 LPBKENB0	Drives the value of USB3 PHY LOOPBACKENB0 signal
8 —	Reserved

Table continues on the next page...

**SCFG\_USB3PRM3CR field descriptions (continued)**

Field	Description
9–15 mPLL_MULT	Drives the value of USB3 mpLL_multiplier signal
16–31 —	Reserved

**12.3.10 USB2 ICID Register (SCFG\_USB2\_ICID)**

The USB2 ICID register contains the bits to provide the ICID for USB2. This register is reset at HRESET.

Address: 157\_0000h base + 100h offset = 157\_0100h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ICID								Lock_Bit							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_USB2\_ICID field descriptions**

Field	Description
0–7 ICID	ICID of USB2 controller (ICID input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.
9–31 —	Reserved

### 12.3.11 USB3 ICID Register (SCFG\_USB3\_ICID)

The USB3 ICID register contains the bits to provide the ICID for USB3. This register is reset at HRESET.

Address: 157\_0000h base + 104h offset = 157\_0104h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_USB3\_ICID field descriptions**

Field	Description
0–7 ICID	ICID of USB3 controller (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.
9–31 —	Reserved

### 12.3.12 qDMA ICID Register (SCFG\_DMA\_ICID)

The qDMA ICID register contains the bits to provide the ICID for qDMA to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 114h offset = 157\_0114h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_DMA\_ICID field descriptions

Field	Description
0–7 ICID	ICID of qDMA (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.
9–31 —	Reserved

### 12.3.13 SATA ICID Register (SCFG\_SATA\_ICID)

The SATA ICID register contains the bits to provide the ICID for SATA to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 118h offset = 157\_0118h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_SATA\_ICID field descriptions**

Field	Description
0–7 ICID	ICID of SATA controller (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.
9–31 —	Reserved

### 12.3.14 USB1 ICID Register (SCFG\_USB1\_ICID)

The USB1 ICID register contains the bits to provide the ICID for USB1. This register is reset at HRESET.

Address: 157\_0000h base + 11Ch offset = 157\_011Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	—	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	—	0	0	0

#### SCFG\_USB1\_ICID field descriptions

Field	Description
0–7 ICID	ICID of USB1 controller (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.  Once this bit is set, write access is disabled to ICID[0:7] bit.
9–31 —	Reserved

### 12.3.15 QE ICID Register (SCFG\_QE\_ICID)

The QE ICID register contains the bits to provide the ICID for QE to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 120h offset = 157\_0120h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_QE\_ICID field descriptions

Field	Description
0–7 ICID	ICID of QE (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.  Once this bit is set, write access is disabled to ICID[0:7] bit.
9–31 —	Reserved

### 12.3.16 eSDHC ICID Register (SCFG\_SDHC\_ICID)

The eSDHC ICID register contains the bits to provide the ICID for eSDHC to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 124h offset = 157\_0124h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_SDHC\_ICID field descriptions

Field	Description
0–7 ICID	ICID of eSDHC (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.  Once this bit is set, write access is disabled to ICID[0:7] bit.
9–31 —	Reserved

### 12.3.17 eDMA ICID Register (SCFG\_eDMA\_ICID)

The eDMA ICID register contains the bits to provide the ICID for eDMA to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 128h offset = 157\_0128h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_eDMA\_ICID field descriptions

Field	Description
0–7 ICID	ICID of eDMA (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.  Once this bit is set, write access is disabled to ICID[0:7] bit.
9–31 —	Reserved

### 12.3.18 ETR ICID Register (SCFG\_ETR\_ICID)

The ETR ICID register contains the bits to provide the ICID for ETR to the SMMU. This register is reset at HRESET

Address: 157\_0000h base + 12Ch offset = 157\_012Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_ETR\_ICID field descriptions

Field	Description
0–7 ICID	ICID of ETR (input to the SMMU).
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.  Once this bit is set, write access is disabled to ICID[0:7] bit.
9–31 —	Reserved

### 12.3.19 Core 0 soft reset Register (SCFG\_CORE0\_SFT\_RST)

The core0 soft reset register contains the bits to provide the soft reset to core0 of A53 complex. This register is reset at HRESET.

Address: 157\_0000h base + 130h offset = 157\_0130h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_CORE0\_SFT\_RST field descriptions

Field	Description
0 soft_reset	Reset process is initiated for core 0 when 1 is written on this bit. The bit is self clearing, and read always get 0. 0 Default 1 Reset core 0 (self-clearing bit)
1–31 —	Reserved

### 12.3.20 Core 1 soft reset Register (SCFG\_CORE1\_SFT\_RST)

The core1 soft reset register contains the bits to provide the soft reset to core1 of A53 complex. This register is reset at HRESET.

Address: 157\_0000h base + 134h offset = 157\_0134h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_CORE1\_SFT\_RST field descriptions

Field	Description
0 soft_reset	Reset process is initiated for core 1 when a 1 is written on this bit. The bit is self clearing, and read always get 0. 0 Default 1 Reset core 1 (self-clearing bit)
1–31 —	Reserved

### 12.3.21 Core 2 soft reset Register (SCFG\_CORE2\_SFT\_RST)

The core2 soft reset register contains the bits to provide the soft reset to core2 of A53 complex. This register is reset at HRESET.

Address: 157\_0000h base + 138h offset = 157\_0138h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_CORE2\_SFT\_RST field descriptions

Field	Description
0 soft_reset	Reset process is initiated for core 2 when a 1 is written on this bit. The bit is self clearing, and read always get 0. 0 Default 1 Reset core 2 (self-clearing bit)
1–31 —	Reserved

### 12.3.22 Core 3soft reset Register (SCFG\_CORE3\_SFT\_RST)

The core3 soft reset register contains the bits to provide the soft reset to core3 of A53 complex. This register is reset at HRESET.

Address: 157\_0000h base + 13Ch offset = 157\_013Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_CORE3\_SFT\_RST field descriptions

Field	Description
0 soft_reset	Reset process is initiated for core 3 when a 1 is written on this bit. The bit is self clearing, and read always get 0. 0 Default 1 Reset core 3 (self-clearing bit)
1–31 —	Reserved

### 12.3.23 PEX PME control register (SCFG\_PEXPMECR)

The PEX PME control register contains the bits to generate PM turnoff message for power management. This register is reset at HRESET.

Address: 157\_0000h base + 144h offset = 157\_0144h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_PEXPMECR field descriptions**

Field	Description
0 PEX1PME	Generates PM turnoff message for power management for PCI Express1. It should be cleared by software.  0 Default 1 RC mode only
1–3 -	This field is reserved.
4 PEX2PME	Generates PM turnoff message for power management for PCI Express2. It should be cleared by software.  0 Default 1 RC mode only
5–7 -	This field is reserved.
8 PEX3PME	Generates PM turnoff message for power management for PCI Express3. It should be cleared by software.  0 Default 1 RC mode only
9–31 -	This field is reserved.

### 12.3.24 FTM chain configuration (SCFG\_FTM\_CHAIN\_CONFIG)

This register contains the bits to enable chaining of FlexTimers. This register is reset at HRESET.

Address: 157\_0000h base + 154h offset = 157\_0154h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	FTM_CHN1	FTM_CHN2	FTM_CHN3	FTM_CHN4	Reserved											
W					Reserved											

#### SCFG\_FTM\_CHAIN\_CONFIG field descriptions

Field	Description
0–15 -	This field is reserved. Reserved
16 FTM_CHN1	FTM_CHN1 0 FTM1 and FTM5 are not chained 1 FTM1 and FTM5 are chained
17 FTM_CHN2	FTM_CHN2 0 FTM2 and FTM6 are not chained 1 FTM2 and FTM6 are chained
18 FTM_CHN3	FTM_CHN3 0 FTM3 and FTM7 are not chained 1 FTM3 and FTM7 are chained
19 FTM_CHN4	FTM_CHN4 0 FTM4 and FTM8 are not chained 1 FTM4 and FTM8 are chained
20–31 -	This field is reserved. Reserved

### 12.3.25 ALTCBAR Register (SCFG\_ALTCBAR)

The ALTCBAR register contains the bits for alternate configuration base address register for PBL. This register is reset at HRESET.

Address: 157\_0000h base + 158h offset = 157\_0158h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0

#### SCFG\_ALTCBAR field descriptions

Field	Description
0–23 ALTCBAR	Alt configuration base address register for PBL.
24–31 —	Reserved

### 12.3.26 QSPI CONFIG Register (SCFG\_QSPI\_CFG)

The QuadSPI configuration register contains the bits for QuadSPI configuration. This register is reset at HRESET.

Address: 157\_0000h base + 15Ch offset = 157\_015Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0

#### SCFG\_QSPI\_CFG field descriptions

Field	Description
0–3 CLK_SEL	These bits control the division of CGA1/CGA2 PLL clock to generate QuadSPI interface clocks. 0000 Divide by 256 0001 Divide by 64 0010 Divide by 32 0011 Divide by 24 0100 Divide by 20 0101 Divide by 16 0110 Divide by 12

*Table continues on the next page...*

**SCFG\_QSPI\_CFG field descriptions (continued)**

Field	Description
	0111 Divide by 8
4–31 —	This field is reserved. Reserved

**12.3.27 QOS1 Register (SCFG\_QOS1)**

The QOS1 register contains the bits for QoS inputs to CCI/interconnect fabric. This register is reset at HRESET. Note that the bits are described as [x:x+3] however are connected to QOS ports as [3:0]. So, there is a bit reversal involved.

Address: 157\_0000h base + 16Ch offset = 157\_016Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	eDMA_QoS	USB2_QoS	USB3_QoS	qDMA_QoS	PEX2_QoS	PEX1_QoS	SEC_QoS	FM_QoS																								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SCFG\_QOS1 field descriptions**

Field	Description
0–3 eDMA_QoS	QOS[3:0] for eDMA.
4–7 USB2_QoS	QOS[3:0] for USB2.
8–11 USB3_QoS	QOS[3:0] for USB3.
12–15 qDMA_QoS	QOS[3:0] for qDMA.
16–19 PEX2_QoS	QOS[3:0] for PCI Express2.
20–23 PEX1_QoS	QOS[3:0] for PCI Express1.
24–27 SEC_QoS	QOS[3:0] for SEC.
28–31 FM_QoS	QOS[3:0] for FMan.

### 12.3.28 QOS2 Register (SCFG\_QOS2)

The QOS2 register contains the bits for QoS inputs to CCI/interconnect fabric. This register is reset at HRESET. Note that the bits are described as [x:x+3]however are connected to QOS ports as [3:0]. So, there is a bit reversal involved.

Address: 157\_0000h base + 170h offset = 157\_0170h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	USB1_QoS	PEX3_QoS	QMan_QoS	A53_QoS	—	eSDHC_QoS	QE_QoS	SATA_QoS																								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### SCFG\_QOS2 field descriptions

Field	Description
0–3 USB1_QoS	QOS[3:0] for USB1.
4–7 PEX3_QoS	QOS[3:0] for PCI Express3.
8–11 QMan_QoS	QOS[3:0] for QMan.
12–15 A53_QoS	QOS[3:0] for A53.
16–19 —	Reserved
20–23 eSDHC_QoS	QOS[3:0] for eSDHC.
24–27 QE_QoS	QOS[3:0] for QE.
28–31 SATA_QoS	QOS[3:0] for SATA.

### 12.3.29 GIC-400 Address 64K Page Alignment Register (SCFG\_GIC400\_ADDR\_ALIGN\_64K)

The GIC-400 address 64 K page alignment register controls the GIC-400 addressing. This register is reset at PORESET.

Address: 157\_0000h base + 188h offset = 157\_0188h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	GIC_ADDR								—							
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_GIC400\_ADDR\_ALIGN\_64K field descriptions

Field	Description
0 GIC_ADDR	Controls the GIC-400 addressing. 0 The addressing of GIC-400 is 64 K page aligned (Default). 1 The GIC-400 addressing is non-64 K page aligned.
1–31 —	Reserved

### 12.3.30 Debug ICID Register (SCFG\_Debug\_ICID)

The debug ICID register contains the bits to provide the ICID for the debug components to the SMMU. This register is reset at HRESET.

Address: 157\_0000h base + 18Ch offset = 157\_018Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_Debug\_ICID field descriptions**

Field	Description
0–7 ICID	ICID input to the SMMU.
8 Lock_Bit	1 Register bits are locked. Any write to update ICID or lock field have no effect. 0 Register bits are open. Write can be done to ICID and lock field.
9–31 —	Reserved

### 12.3.31 Snoop Configuration Register (SCFG\_SNPCNFGCR)

The snoop configuration control register contains the bits to drive snoop signal for various masters. This register is reset at HRESET.

Address: 157\_0000h base + 1A4h offset = 157\_01A4h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SECRDSNP	SECWRSNP						—	SATARD SNP	SATAWR SNP	USB1RDSNP	USB1WR SNP	DBGRD SNP	DBGWWR SNP	eDMASNP	USB2WR SNP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	USB2RDSNP	USB3WRSNP	USB3RDSNP						—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_SNPCNFGCR field descriptions

Field	Description
0 SECRDSNP	Drives SEC Read snoop signal. 0 SEC reads are not snoopable 1 SEC reads are snoopable
1 SECWRSNP	Controls snoop attribute of SEC writes. 0 SEC writes are not snoopable 1 SEC writes are snoopable
2–7 —	Reserved
8 SATARD SNP	Drives SATA read snoop signal. 0 SATA reads are not snoopable 1 SATA reads are snoopable
9 SATAWR SNP	Controls snoop attribute of SATA writes. 0 SATA writes are not snoopable 1 SATA writes are snoopable

Table continues on the next page...

**SCFG\_SNPCNFGCR field descriptions (continued)**

Field	Description
10 USB1RDSNP	Drives USB1 read snoop signal 0 USB1 reads are not snoopable 1 USB1 reads are snoopable
11 USB1WRSNP	Controls snoop attribute of USB3 writes. 0 USB1 writes are not snoopable 1 USB1 writes are snoopable
12 DBGRDSNP	0 Debug reads are not snoopable 1 Debug reads are snoopable
13 DBGWRSNP	Drives DEBUG (debug and etr port of interconnect fabric) Write Snoop signal. Controls snoop attribute of Debug writes. 0 Debug writes are not snoopable 1 Debug writes are snoopable
14 eDMASNP	Drives eDMA snoop signal. If eDMA is used for SPI, I2C, QuadSPI, FlexTimer, and LPUART then this bit configures the snooping attribute for all the transactions (R/W) initiated by eDMA on the behalf of these IPs. 0 eDMA transactions are not snoopable 1 eDMA transactions are snoopable
15 USB2WRSNP	Drives USB2 write snoop signal. 0 USB2 writes are not snoopable 1 USB2 writes are snoopable
16 USB2RDSNP	Drives USB2 read snoop signal. 0 USB2 reads are not snoopable 1 USB2 reads are snoopable
17 USB3WRSNP	Drives USB3 write snoop signal. 0 USB3 writes are not snoopable 1 USB3 writes are snoopable
18 USB3RDSNP	Drives USB3 read snoop signal. 0 USB3 reads are not snoopable 1 USB3 reads are snoopable
19–31 —	Reserved

### 12.3.32 Interrupt Polarity Register (SCFG\_INTPCR)

The interrupt polarity control register contains the bits to control the polarity of the IRQ0-11. This register is reset at HRESET.

Address: 157\_0000h base + 1ACh offset = 157\_01ACh

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	IRQ0INTP	IRQ1INTP	IRQ2INTP	IRQ3INTP	IRQ4INTP	IRQ5INTP	IRQ6INTP	IRQ7INTP	IRQ8INTP	IRQ9INTP	IRQ10INTP	IRQ11INTP	—	—	—	—
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_INTPCR field descriptions

Field	Description
0 IRQ0INTP	Controls the polarity of IRQ0 0 IRQ0 is active high 1 IRQ0 is active low
1 IRQ1INTP	Controls the polarity of IRQ1 0 IRQ1 is active high 1 IRQ1 is active low
2 IRQ2INTP	Controls the polarity of IRQ2 0 IRQ2 is active high 1 IRQ2 is active low
3 IRQ3INTP	Controls the polarity of IRQ3 0 IRQ3 is active high 1 IRQ3 is active low
4 IRQ4INTP	Controls the polarity of IRQ4 0 IRQ4 is active high 1 IRQ4 is active low
5 IRQ5INTP	Controls the polarity of IRQ5 0 IRQ5 is active high 1 IRQ5 is active low

Table continues on the next page...

**SCFG\_INTPCR field descriptions (continued)**

Field	Description
6 IRQ6INTP	Controls the polarity of IRQ6 0 IRQ6 is active high 1 IRQ6 is active low
7 IRQ7INTP	Controls the polarity of IRQ7 0 IRQ7 is active high 1 IRQ7 is active low
8 IRQ8INTP	Controls the polarity of IRQ8 0 IRQ8 is active high 1 IRQ8 is active low
9 IRQ9INTP	Controls the polarity of IRQ9 0 IRQ9 is active high 1 IRQ9 is active low
10 IRQ10INTP	Controls the polarity of IRQ10 0 IRQ10 is active high 1 IRQ10 is active low
11 IRQ11INTP	Controls the polarity of IRQ11 0 IRQ11 is active high 1 IRQ11 is active low
12–31 —	Reserved

### 12.3.33 CORE Soft Reset Enable Register (SCFG\_CORESRENCR)

The core soft reset enable control register contains the bit to enable the core soft reset functionality. This register is reset at PORESET.

Address: 157\_0000h base + 204h offset = 157\_0204h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_CORESRENCR field descriptions

Field	Description
0 CORESREN	Controls the enable of core soft reset functionality 0 Core soft-reset is disabled(Default) 1 Core soft-reset is enabled
1–31 —	Reserved

### 12.3.34 Core 0 Reset Vector Base Address0 (SCFG\_RVBAR0\_0)

The core 0 reset vector base address0 register controls the reset vector base address for core 0 for bits [33:2]. This register is reset at PORESET. This register should be programmed in the PBI phase.

#### NOTE

After core warm reset, the core can boot from a reset vector configured in SCFG\_RVBAR0\_0 (Core 0 Reset Vector Base Address0) and SCFG\_RVBARn\_1 (Core 0 Reset Vector Base Address1).

The 40-b address should be programmed in RVBARn\_0/ RVBARn\_1 register for the entry point of the vector from where the core executes the first instruction after coming out from the warm-reset. If it is not configured, its reset value will be 0x0 and the first instruction will be executed from the internal boot ROM.

Address: 157\_0000h base + 220h offset = 157\_0220h

## SCFG RVBAR0 0 field descriptions

Field	Description
0–31 RVBAR0_0	Core 0 reset vector base address for bits [33:2].

### 12.3.35 Core 0 Reset Vector Base Address1 (SCFG\_RVBAR0\_1)

The core 0 reset vector base address1 register controls the reset vector base address for core 0 for bits [39:34]. This register is reset at PORESET. This register should be programmed in the PBI phase.

## **NOTE**

After core warm reset, the core can boot from a reset vector configured in SCFG\_RVBAR0\_0 (Core 0 Reset Vector Base Address0) and SCFG\_RVBARn\_1 (Core 0 Reset Vector Base Address1).

The 40-b address should be programmed in RVBARn\_0/RVBARn\_1 register for the entry point of the vector from where the core executes the first instruction after coming out from the warm-reset. If it is not configured, its reset value will be 0x0 and the first instruction will be executed from the internal boot ROM.

Address: 157 0000h base + 224h offset = 157 0224h

## SCFG RVBAR0 1 field descriptions

Field	Description
0–5 RVBAR0_1	Core 0 reset vector base address for bits [39:34].
6–31 —	Reserved

### 12.3.36 Core 1 Reset Vector Base Address0 (SCFG RVBAR1 0)

The core 1 reset vector base address0 register controls the reset vector base address for core 1 for bits [33:2]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157 0000h base + 228h offset = 157 0228h

**SCFG RVBAR1 0 field descriptions**

Field	Description
0–31 RVBAR1_0	Core 1 reset vector base address for bits [33:2].

### 12.3.37 Core 1 Reset Vector Base Address1 (SCFG\_RVBAR1\_1)

The RVBAR1\_1 register controls the reset vector base address for core 1 for bits [39:34]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157\_0000h base + 22Ch offset = 157\_022Ch

## SCFG RVBAR1 1 field descriptions

Field	Description
0–5 RVBAR1_1	Core 1 reset vector base address for bits [39:34].

*Table continues on the next page...*

**SCFG\_RVBAR1\_1 field descriptions (continued)**

Field	Description
6–31 —	Reserved

**12.3.38 Core 2 Reset Vector Base Address0 (SCFG\_RVBAR2\_0)**

The core 2 reset vector base address0 register controls the reset vector base address for core 0 for bits [33:2]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157\_0000h base + 230h offset = 157\_0230h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																																	
W																																	

Reset 0

**SCFG\_RVBAR2\_0 field descriptions**

Field	Description
0–31 RVBAR2_0	Core 2 reset vector base address for bits [33:2].

**12.3.39 Core 2 Reset Vector Base Address1 (SCFG\_RVBAR2\_1)**

The core 2 reset vector base address1 register controls the reset vector base address for core 2 for bits [39:34]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157\_0000h base + 234h offset = 157\_0234h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																																	
W																																	

Reset 0

**SCFG\_RVBAR2\_1 field descriptions**

Field	Description
0–5 RVBAR2_1	Core 2 reset vector base address for bits [39:34].

*Table continues on the next page...*

**SCFG\_RVBAR2\_1 field descriptions (continued)**

Field	Description
6–31 —	Reserved

**12.3.40 Core 3 Reset Vector Base Address0 (SCFG\_RVBAR3\_0)**

The core 3 reset vector base address0 controls the reset vector base address for core 3 for bits [33:2]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157\_0000h base + 238h offset = 157\_0238h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																																	
W																																	

Reset 0

**SCFG\_RVBAR3\_0 field descriptions**

Field	Description
0–31 RVBAR3_0	Core 3 reset vector base address for bits [33:2].

**12.3.41 Core 3 Reset Vector Base Address1 (SCFG\_RVBAR3\_1)**

The core 3 reset vector base address1 register controls the reset vector base address for core 3 for bits [39:34]. This register is reset at PORESET. This register should be programmed in the PBI phase.

Address: 157\_0000h base + 23Ch offset = 157\_023Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0

**SCFG\_RVBAR3\_1 field descriptions**

Field	Description
0–5 RVBAR3_1	Core 3 reset vector base address for bits [39:34].

*Table continues on the next page...*

**SCFG\_RVBAR3\_1 field descriptions (continued)**

Field	Description
6–31 —	Reserved

### 12.3.42 Core Low Power Mode Control Status Register (SCFG\_LPMCSR)

The LPMCSR register provides status and control bits for various signals on A53. This register is reset at PORESET.

Address: 157\_0000h base + 240h offset = 157\_0240h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	—	—	—	SMPEN3	CPUQDENY3	CPUQACCEPTn <sub>3</sub>	CPUQACTIVE3	CPUQREQn3	—	—	—	SMPEN2	CPUQDENY2	CPUQACCEPTn <sub>2</sub>	CPUQACTIVE2	CPUQREQn2
W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—	—	—	SMPEN1	CPUQDENY1	CPUQACCEPTn <sub>1</sub>	CPUQACTIVE1	CPUQREQn1	—	—	—	SMPENO	CPUQDENYO	CPUQACCEPTn <sub>0</sub>	CPUQACTIVE0	CPUQREQn0
W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

**SCFG\_LPMCSR field descriptions**

Field	Description
0–2 —	Reserved
3 SMPEN3	Status bit for SMPEN signal of core 3.
4 CPUQDENY3	Status bit for CPUQDENY signal for core 3.
5 CPUQACCEPTn3	Status bit for CPUQACCEPTn signal for core 3.
6 CPUQACTIVE3	Status bit for CPUQACTIVE signal for core 3.

Table continues on the next page...

**SCFG\_LPMCSR field descriptions (continued)**

Field	Description
7 CPUQREQn3	Control bit for CPUQREQn for core 3.
8–10 —	Reserved
11 SMPEN2	Status bit for SMPEN signal of core 2.
12 CPUQDENY2	Status bit for CPUQDENY signal for core 2.
13 CPUQACCEPTn2	Status bit for CPUQACCEPTn signal for core 2.
14 CPUQACTIVE2	Status bit for CPUQACTIVE signal for core 2.
15 CPUQREQn2	Control bit for CPUQREQn for core 2.
16–18 —	Reserved
19 SMPEN1	Status bit for SMPEN signal of core 1.
20 CPUQDENY1	Status bit for CPUQDENY signal for core 1.
21 CPUQACCEPTn1	Status bit for CPUQACCEPTn signal for core 1.
22 CPUQACTIVE1	Status bit for CPUQACTIVE signal for core 1.
23 CPUQREQn1	Control bit for CPUQREQn for core 1.
24–26 —	Reserved
27 SMPEN0	Status bit for SMPEN signal of core 0.
28 CPUQDENY0	Status bit for CPUQDENY signal for core 0.
29 CPUQACCEPTn0	Status bit for CPUQACCEPTn signal for core 0.
30 CPUQACTIVE0	Status bit for CPUQACTIVE signal for core 0.
31 CPUQREQn0	Control bit for CPUQREQn for core 0.

### 12.3.43 ECGTX Clock Mux Control Register (SCFG\_ECGTXCMCR)

The ECGTX clock mux control register contains the bits to support FMan clock multiplexing. This register is reset on PORESET.

Address: 157\_0000h base + 404h offset = 157\_0404h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	—	—	—	—	CLK_SEL	—	—	—	—	—	—	—	—	—	—	—
W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_ECGTXCMCR field descriptions

Field	Description
0–3 —	Reserved
4 CLK_SEL	Selects 125MHz reference clock for RGMII. 0 EC1_GTX_CLK125 is used as clock source 1 EC2_GTX_CLK125 is used as clock source
5–31 —	Reserved

### 12.3.44 SDHC IO VSEL Control Register (SCFG\_SDHCIOVSELCR)

The SDHC IO VSEL control register contains the bits to support SDHC IO voltage switching. This register is reset on HRESET.

Address: 157\_0000h base + 408h offset = 157\_0408h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TGLEN		VSELVAL						—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SCFG\_SDHCIOVSELCR field descriptions

Field	Description
0 TGLEN	SDHC IO voltage switching enable 0 Voltage switching not enabled (default) 1 Voltage switching enabled.
1–2 VSELVAL	Configures voltage for SDHC ( if enabled by TGLEN) 0b00 - 1.8V 0b01 - Reserved 0b10 - 3.3V 0b11 - Auto-voltage-selection enabled
3–30 —	Reserved
31 SDHC_VS	SCFG bit reflecting shadow bit controlled by switch for SDHC:VOLT_SEL 0 Change the SD bus supply voltage to high voltage range, 3.3V 1 Change the SD bus supply voltage to low voltage range, 1.8V

### 12.3.45 Extended RCW PinMux Control Register (SCFG\_RCWPMUXCR0)

The extended RCW controlled pinmux register contains the bits to provide bits for pin multiplexing control. This register is reset on HRESET.

Address: 157\_0000h base + 40Ch offset = 157\_040Ch

Bit	0	1	2	3	4	5	6	7	—	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	—	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	—	24	25	26	27	28	29	30	31
Reset	0	0	0	0	0	0	0	0	—	0	0	0	0	0	0	0	0

**SCFG\_RCWPMUXCR0 field descriptions**

Field	Description
0–16 —	Reserved
17–19 IIC3_SCL	Configures functionality of the IIC3_SCL. Options: 000 IIC3_SCL 001 GPIO_4[10] 010 EVT_B[5] 011 USB2_DRVVBUS 100 BRGO4 101 FTM8_CH0 110 CLK11 111 Reserved
20 —	Reserved
21–23 IIC3_SDA	Configures functionality of the IIC3_SDA Options: 000 IIC3_SDA 001 GPIO_4[11] 010 EVT_B[6] 011 USB2_PWRFAULT 100 BRGO1 101 FTM8_CH1

*Table continues on the next page...*

**SCFG\_RCWPMUXCR0 field descriptions (continued)**

Field	Description
	110 CLK12_CLK8 (2 pins of QE are connected) 111 Reserved
24 —	Reserved
25–27 IIC4_SCL	Configures functionality of the IIC4_SCL Options: 000 IIC4_SCL 001 GPIO_4[12] 010 EVT_B[7] 011 USB3_DRVVBUS 100 TDMA_RQ 101 FTM3_FAULT 110 UC1_CDB_RXER 111 Reserved Refer USB DRVVBUS Select Register (USB_DRVVBUS_SELCR) for USB3_DRV_VBUS to USB controller mapping.
28 —	Reserved
29–31 IIC4_SDA	Configures functionality of the IIC4_SDA Options: 000 IIC4_SDA 001 GPIO_4[13] 010 EVT_B[8] 011 USB3_PWRFAULT 100 TDMB_RQ 101 FTM3_EXTCLK 110 UC3_CDB_RXER 111 IIC4_SDA Refer USB PWRFAULT Select Register (USB_PWRFAULT_SELCR) for USB3_PWRFAULT to USB controller mapping.

### **12.3.46 USB DRVVBUS Control Register (SCFG\_USBDRVVBUS\_SELCR)**

The USB DRVVBUS select register contains the bits to provide control the USB which drives  $USBn\_DRVVBUS$ . This register is reset on HRESET.

Address: 157\_0000h base + 410h offset = 157\_0410h

## SCFG USBDRVVBUS SELCR field descriptions

Field	Description
0-29 —	Reserved
30-31 USB_SEL	Selection of USB Controller to drive USB_DRVVBUS I/O Options: 00 USB_DRVVBUS I/O driven by USB Controller 1 01 USB_DRVVBUS I/O driven by USB Controller 2 10 Reserved 11 USB_DRVVBUS I/O driven by USB Controller 3

### **12.3.47 USB PWRFAULTControl Register (SCFG\_USBPOWERFAULT\_SELCR)**

The USB PWRFAULT select register defines how USB\_PWR\_FAULT is sampled by all three controllers. This register is reset on HRESET.

Address: 157 0000h base + 414h offset = 157 0414h

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R									—								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R									—								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**SCFG\_USBWRFAULT\_SELCR field descriptions**

Field	Description
0–25 —	Reserved
26–27 USB3_SEL	USB controller 3 connectivity Option(s): 00 USB controller 3 has its PWRFAULT input tied inactive (no fault) 01 USB controller 3 receives PWRFAULT from shared USB_PWRFAULT I/O. 10 USB controller 3 receives PWRFAULT from dedicated USB3_PWRFAULT I/O 11 Reserved
28–29 USB2_SEL	USB controller 2 connectivity Option(s): 00 USB controller 2 has its PWRFAULT input tied inactive (no fault) 01 USB controller 2 receives PWRFAULT from shared USB_PWRFAULT I/O. 10 USB controller 2 receives PWRFAULT from dedicated USB2_PWRFAULT I/O 11 Reserved
30–31 USB1_SEL	USB Controller 1 connectivity Option(s): 00 USB controller 1 has its PWRFAULT input tied inactive (no fault) 01 USB controller 1 receives PWRFAULT from shared USB_PWRFAULT I/O. 10 Reserved 11 Reserved

### 12.3.48 USB PHY1 Reference Clock Select Register (SCFG\_USB\_REFCLK\_SELCR1)

The USB PHY1 reference clock select register contains bits to select the reference clock for USB PHY1. This register is reset on HRESET.

Address: 157\_0000h base + 418h offset = 157\_0418h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	RST								—							
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W				—										FSEL		CKSEL
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0

**SCFG\_USB\_REFCLK\_SELCR1 field descriptions**

Field	Description
0 RST	Active high reset. 0 Reset is de-asserted 1 Reset is asserted
1–23 —	Reserved
24–29 FSEL	Controls USB PHY PLL reference clock frequency. Recommended value is 5'b100111 that corresponds to 100 MHz.
30–31 CKSEL	Selects the reference clock for USB PHY 1 PLL. 00 SYSCLK 10 DIFF_SYSCLK/DIFF_SYSCLK_B

**12.3.49 USB PHY2 Reference Clock Select Register (SCFG\_USB\_REFCLK\_SELCR2)**

The The USB PHY2 reference clock select register contains bits to select the reference clock for USB PHY2. This register is reset on HRESET.

Address: 157\_0000h base + 41Ch offset = 157\_041Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	RST								—							
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W			—											FSEL		CKSEL
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0

**SCFG\_USB\_REFCLK\_SELCR2 field descriptions**

Field	Description
0 RST	Active high reset. 0 Reset is de-asserted 1 Reset is asserted
1–23 —	Reserved
24–29 FSEL	Controls USB PHY PLL reference clock frequency.

*Table continues on the next page...*

**SCFG\_USB\_REFCLK\_SELCR2 field descriptions (continued)**

Field	Description
	Recommended value is 5'b100111 that corresponds to 100 MHz.
30–31 CKSEL	Selects the reference clock for USB PHY 2 PLL 00 SYSCLK 10 DIFF_SYSCLK/DIFF_SYSCLK_B

### 12.3.50 USB PHY3 Reference Clock Select Register (SCFG\_USB\_REFCLK\_SELCR3)

The USB PHY3 reference clock select register contains bits to select the reference clock for USB PHY3. This register is reset on HRESET.

Address: 157\_0000h base + 420h offset = 157\_0420h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	RST								—							
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W			—												CKSEL	
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0

**SCFG\_USB\_REFCLK\_SELCR3 field descriptions**

Field	Description
0 RST	Active high reset. 0 Reset is de-asserted 1 Reset is asserted
1–23 —	Reserved
24–29 FSEL	Controls USB PHY PLL reference clock frequency. Recommended value is 5'b100111 that corresponds to 100 MHz.
30–31 CKSEL	Selects the reference clock for USB PHY 3 PLL 00 SYSCLK 10 DIFF_SYSCLK/DIFF_SYSCLK_B

### 12.3.51 Retention Request Control Register (SCFG\_RETREQCR)

The RETREQCR register contains the bits to enable retention request. This register is reset on HRESET.

Address: 157\_0000h base + 424h offset = 157\_0424h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RETREQ0	RETREQ1	RETREQ2	RETREQ3												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_RETREQCR field descriptions**

Field	Description
0 RETREQ0	Retention request enable 0.
1 RETREQ1	Retention request enable 1.
2 RETREQ2	Retention request enable 2.
3 RETREQ3	Retention request enable 3.
4–31 —	Reserved

### 12.3.52 CORE PM Control Register (SCFG\_COREPMCR)

The COREPMCR register contains control bit to enable WFIL2. This register is reset on HRESET.

Address: 157\_0000h base + 42Ch offset = 157\_042Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									—							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_COREPMCR field descriptions**

Field	Description
0–30 —	Reserved
31 WFIL2EN	WFIL2 Enable for core A53.

### 12.3.53 SCRATCHRWn - Scratch Read Write Registers (SCFG\_SCRATCHRWn)

The SCRATCHn read write register n(n=1 to 4), provides read/write scratch register locations available to the user. It provides expansion bits for device control. This register is reset on HRESET.

Note that the SCRATCHRW1 and SCRATCHRW2 registers are used for non-secure boot whereas SCRATCHRW3 and SCRATCHRW4 registers are defined for secure-boot.

When performing non-secure boot, these registers are defined as follows:

SCRATCHRW1 - Reserved for future use in case boot location pointer is 64 bit. The value must be 0.

## SCRATCHRW2 - 32-bit boot location pointer (BOOTLOCPTR)

When performing secure boot, these registers are defined as follows:

SCRATCHRW3 - Register that indicate failures in SEC self tests (if any) conducted as part of secure boot flow

SCRATCHRW4- Register to control setting of ClientPD in SMMU at IBR exit.

- SFP\_OSPR(ITS) = 1, Register is don't care and ClientPD bit is always left set as 0
- SFP\_OSPR(ITS)ITS = 0, SB\_EN = 1
  - SCRATCHRW4 = 0 (default), ClientPD is left set as 0
  - SCRATCHRW4 = non-zero, ClientPD is reset to 1 (by-pass mode)

Address: 157\_0000h base + 600h offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																VAL																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### SCFG\_SCRATCHRWn field descriptions

Field	Description
0–31 VAL	32-bit scratch contents

## 12.3.54 Core Boot Control Register (SCFG\_COREBCR)

The COREBCR register provides expansion bits for device control. This register is reset on HRESET. The bits get set on assertion of core reset.

Address: 157\_0000h base + 680h offset = 157\_0680h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

**SCFG\_COREBCR field descriptions**

Field	Description
0–27 —	Reserved
28 CORE3	Write 1 to clear bit for core 3. Also set with core 3 reset.
29 CORE2	Write 1 to clear bit for core 2. Also set with core 2 reset
30 CORE1	Write 1 to clear bit for core 1. Also set with core 1 reset.
31 CORE0	Write 1 to clear bit for core 0. Also set with core 0 reset.

**12.3.55 Shared Message Signaled Interrupt Index Register (SCFG\_G0MSIIR)**

The group 0 (G0) shared message signaled interrupt index register provides the mechanism for setting an interrupt in the MSIR. When MSIIR is written, MSIIR[IBS] selects the shared interrupt field in the selected MSIR register. MSIIR is primarily intended to support PCI express MSIs.

This register can be written by any of the PCI Express Endpoint device.

Address: 157\_0000h base + 1000h offset = 157\_1000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—			IBS		SRS											—															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SCFG\_G0MSIIR field descriptions**

Field	Description
0–2 —	Reserved
3–5 IBS	Interrupt bit selects the bit of the chosen logical section to be set in the GnMSIRx.  MSIR1  000 Set field SH0 (bit 0) 001 Set field SH1 (bit 1) 010 Set field SH2 (bit 2) 011 Set field SH3 (bit 3) 100 Set field SH4 (bit 4) 101 Set field SH5 (bit 5) 110 Set field SH6 (bit 6) 111 Set field SH7 (bit 7)

Table continues on the next page...

**SCFG\_G0MSIIR field descriptions (continued)**

Field	Description
	MSIR2 000 Set field SH8 (bit 8) 001 Set field SH9 (bit 9) 010 Set field SH10 (bit 10) 011 Set field SH11 (bit 11) 100 Set field SH12 (bit 12) 101 Set field SH13 (bit 13) 110 Set field SH14 (bit 114) 111 Set field SH15 (bit 15) MSIR3 000 Set field SH16 (bit 16) 001 Set field SH17 (bit 17) 010 Set field SH18 (bit 18) 011 Set field SH19 (bit 19) 100 Set field SH20 (bit 20) 101 Set field SH21 (bit 21) 110 Set field SH22 (bit 22) 111 Set field SH23 (bit 23) MSIR4 000 Set field SH24 (bit 24) 001 Set field SH25 (bit 25) 010 Set field SH26 (bit 26) 011 Set field SH27 (bit 27) 100 Set field SH28 (bit 28) 101 Set field SH29 (bit 29) 110 Set field SH30 (bit 30) 111 Set field SH31 (bit 31)
6–7 SRS	Shared interrupt register select. Selects the MSIR to be written. 00 MSIR 1 01 MSIR 2 10 MSIR 3 11 MSIR 4
8–31 —	Reserved

### 12.3.56 Shared Message Signaled Interrupt Register (SCFG\_G0MSIR1)

The group 0 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 1010h offset = 157\_1010h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SHn								—																							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### SCFG\_G0MSIR1 field descriptions

Field	Description
0–7 SHn	Message sharer n has a pending interrupt.
8–31 —	Reserved

### 12.3.57 Shared Message Signaled Interrupt Register (SCFG\_G0MSIR2)

The group 0(G0) shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 1014h offset = 157\_1014h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—							SHn																								
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### SCFG\_G0MSIR2 field descriptions

Field	Description
0–7 —	Reserved
8–15 SHn	Message sharer n has a pending interrupt.

Table continues on the next page...

## **SCFG\_G0MSIR2 field descriptions (continued)**

Field	Description
16–31 —	Reserved

### **12.3.58 Shared Message Signaled Interrupt Register (SCFG\_G0MSIR3)**

The group 0(G0) shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 1018h offset = 157\_1018h

## SCFG G0MSIR3 field descriptions

Field	Description
0–15 —	Reserved
16–23 SHn	Message sharer n has a pending interrupt.
24–31 —	Reserved

### **12.3.59 Shared Message Signaled Interrupt Register (SCFG\_G0MSIR4)**

The group 0(G0) shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 101Ch offset = 157\_101Ch

**SCFG\_G0MSIR4 field descriptions**

Field	Description
0–23 —	Reserved
24–31 SHn	Message sharer n has a pending interrupt.

### 12.3.60 Shared Message Signaled Interrupt Index Register (SCFG\_G1MSIIR)

The G1 shared message signaled interrupt index register provides the mechanism for setting an interrupt in the MSIR. When MSIIR is written, MSIIR[IBS] selects the shared interrupt field in the selected MSIR register. This register is primarily intended to support PCI Express MSIs.

This register can be written by any of the PCI Express Endpoint device.

Address: 157\_0000h base + 2000h offset = 157\_2000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—				IBS		SRS										—															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SCFG\_G1MSIIR field descriptions**

Field	Description
0–2 —	Reserved
3–5 IBS	Interrupt bit selects the bit of the chosen logical section to be set in the GnMSIRx.  MSIR1 000 Set field SH0 (bit 0) 001 Set field SH1 (bit 1) 010 Set field SH2 (bit 2) 011 Set field SH3 (bit 3) 100 Set field SH4 (bit 4) 101 Set field SH5 (bit 5) 110 Set field SH6 (bit 6) 111 Set field SH7 (bit 7)  MSIR2 000 Set field SH8 (bit 8) 001 Set field SH9 (bit 9) 010 Set field SH10 (bit 10)

*Table continues on the next page...*

**SCFG\_G1MSIIR field descriptions (continued)**

Field	Description
	011 Set field SH11 (bit 11) 100 Set field SH12 (bit 12) 101 Set field SH13 (bit 13) 110 Set field SH14 (bit 114) 111 Set field SH15 (bit 15)  MSIR3 000 Set field SH16 (bit 16) 001 Set field SH17 (bit 17) 010 Set field SH18 (bit 18) 011 Set field SH19 (bit 19) 100 Set field SH20 (bit 20) 101 Set field SH21 (bit 21) 110 Set field SH22 (bit 22) 111 Set field SH23 (bit 23)  MSIR4 000 Set field SH24 (bit 24) 001 Set field SH25 (bit 25) 010 Set field SH26 (bit 26) 011 Set field SH27 (bit 27) 100 Set field SH28 (bit 28) 101 Set field SH29 (bit 29) 110 Set field SH30 (bit 30) 111 Set field SH31 (bit 31)
6–7 SRS	Shared interrupt register select. Selects the MSIR to be written.  00 MSIR 1 01 MSIR 2 10 MSIR 3 11 MSIR 4
8–31 —	Reserved

## 12.3.61 Shared Message Signaled Interrupt Register (SCFG\_G1MSIR1)

The group 1 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157 0000h base + 2010h offset = 157 2010h

## SCFG G1MSIR1 field descriptions

Field	Description
0–7 SHn	Message sharer n has a pending interrupt.
8–31 —	Reserved

## 12.3.62 Shared Message Signaled Interrupt Register (SCFG\_G1MSIR2)

The group 1 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157 0000h base + 2014h offset = 157 2014h

## SCFG G1MSIR2 field descriptions

Field	Description
0–7 —	Reserved
8–15 SHn	Message sharer n has a pending interrupt.

*Table continues on the next page...*

**SCFG\_G1MSIR2 field descriptions (continued)**

Field	Description
16–31 —	Reserved

### 12.3.63 Shared Message Signaled Interrupt Register (SCFG\_G1MSIR3)

The group 1 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 2018h offset = 157\_2018h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R									—										SHn									—					
W																																	

**SCFG\_G1MSIR3 field descriptions**

Field	Description
0–15 —	Reserved
16–23 SHn	Message sharer n has a pending interrupt.
24–31 —	Reserved

### 12.3.64 Shared Message Signaled Interrupt Register (SCFG\_G1MSIR4)

The group 1 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 201Ch offset = 157\_201Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																			—									SHn				
W																																

**SCFG\_G1MSIR4 field descriptions**

Field	Description
0–23 —	Reserved
24–31 SHn	Message sharer n has a pending interrupt.

### 12.3.65 Shared Message Signaled Interrupt Index Register (SCFG\_G2MSIIR)

The G2 shared message signaled interrupt index register provides the mechanism for setting an interrupt in the MSIR. When MSIIR is written, MSIIR[IBS] selects the shared interrupt field in the selected MSIR register. MSIIR is primarily intended to support PCI Express MSIs.

This register can be written by any of the PCI Express Endpoint device.

Address: 157\_0000h base + 3000h offset = 157\_3000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	—				IBS		SRS										—															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SCFG\_G2MSIIR field descriptions**

Field	Description
0–2 —	Reserved
3–5 IBS	Interrupt bit selects the bit of the chosen logical section to be set in the GnMSIRx.  MSIR1 000 Set field SH0 (bit 0) 001 Set field SH1 (bit 1) 010 Set field SH2 (bit 2) 011 Set field SH3 (bit 3) 100 Set field SH4 (bit 4) 101 Set field SH5 (bit 5) 110 Set field SH6 (bit 6) 111 Set field SH7 (bit 7)  MSIR2 000 Set field SH8 (bit 8) 001 Set field SH9 (bit 9) 010 Set field SH10 (bit 10)

*Table continues on the next page...*

**SCFG\_G2MSIIR field descriptions (continued)**

Field	Description
	011 Set field SH11 (bit 11) 100 Set field SH12 (bit 12) 101 Set field SH13 (bit 13) 110 Set field SH14 (bit 114) 111 Set field SH15 (bit 15)  MSIR3 000 Set field SH16 (bit 16) 001 Set field SH17 (bit 17) 010 Set field SH18 (bit 18) 011 Set field SH19 (bit 19) 100 Set field SH20 (bit 20) 101 Set field SH21 (bit 21) 110 Set field SH22 (bit 22) 111 Set field SH23 (bit 23)  MSIR4 000 Set field SH24 (bit 24) 001 Set field SH25 (bit 25) 010 Set field SH26 (bit 26) 011 Set field SH27 (bit 27) 100 Set field SH28 (bit 28) 101 Set field SH29 (bit 29) 110 Set field SH30 (bit 30) 111 Set field SH31 (bit 31)
6–7 SRS	Shared interrupt register select. Selects the MSIR to be written.  00 MSIR 1 01 MSIR 2 10 MSIR 3 11 MSIR 4
8–31 —	Reserved

## 12.3.66 Shared Message Signaled Interrupt Register (SCFG\_G2MSIR1)

The group 2 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157 0000h base + 3010h offset = 157 3010h

## SCFG G2MSIR1 field descriptions

Field	Description
0–7 SHn	Message sharer n has a pending interrupt.
8–31 —	Reserved

## 12.3.67 Shared Message Signaled Interrupt Register (SCFG\_G2MSIR2)

The group 2 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157 0000h base + 3014h offset = 157 3014h

## SCFG G2MSIR2 field descriptions

Field	Description
0–7 —	Reserved
8–15 SHn	Message sharer n has a pending interrupt.

*Table continues on the next page...*

**SCFG\_G2MSIR2 field descriptions (continued)**

Field	Description
16–31 —	Reserved

### 12.3.68 Shared Message Signaled Interrupt Register (SCFG\_G2MSIR3)

The group 2 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 3018h offset = 157\_3018h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R									—										SHn									—					
W																																	

**SCFG\_G2MSIR3 field descriptions**

Field	Description
0–15 —	Reserved
16–23 SHn	Message sharer n has a pending interrupt.
24–31 —	Reserved

### 12.3.69 Shared Message Signaled Interrupt Register (SCFG\_G2MSIR4)

The group 2 shared message signaled interrupt register indicates which of the up to 8 interrupt sources sharing a message register have pending interrupts. This register is cleared when read; write to the register has no effect.

Address: 157\_0000h base + 301Ch offset = 157\_301Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																			—										SHn				
W																																	

**SCFG\_G2MSIR4 field descriptions**

Field	Description
0–23 —	Reserved
24–31 SHn	Message sharer n has a pending interrupt.

# Chapter 13

## Device Configuration and Pin Control

### 13.1 Device Configuration and Pin Control Introduction

This chapter describes the device configuration and pin control facilities of the chip.

The device configuration unit provides general purpose configuration and status for the device and the pin control block implements general purpose configuration and status registers for the device, and the logic to configure the I/O pads to operate according to the requirements of the functional components. It also controls the data path between the SoC components and the I/O pads. The pin control block consists of the pins control module, the functional I/O multiplexing, and the JTAG boundary scan logic.

### 13.2 Features

The device configuration unit features the following:

- Pin sampling of device configuration pins at power-on reset and a corresponding POR status register for capturing the values of these configuration pins
- Reset Configuration Word (RCW) support via a set of RCW status registers written by the Preboot Loader (PBL) during power-on or hard reset in the PBL's RCW stage
- Boot release registers(s) used for releasing cores for booting
- Register file for the Reset module including:
  - Register for initiating a device RESET\_REQ\_B through software
  - Set of registers for control and status of sources on the device which can drive the device's RESET\_REQ\_B pin
- Core and device disable registers used for gating off clocks for any IP blocks or cores which are not used at all by an application
- Two small sets of scratch registers:
  - One set of read / write scratch registers
  - One set of write-once / read scratch registers

## 13.3 Device Configuration/Pin Control Memory Map

The table below shows the memory-mapped CCSR registers of the Device Config module and lists the offset, name, and a cross-reference to the complete description of each register. These registers only support 32-bit accesses.

**DCFG\_CCSR memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
1EE_0000	POR Status Register 1 (DCFG_CCSR_PORSR1)	32	R	<a href="#">See section</a>	13.3.1/485
1EE_0004	POR Status Register 2 (DCFG_CCSR_PORSR2)	32	R	<a href="#">See section</a>	13.3.2/486
1EE_0020	General-Purpose POR Configuration Register (DCFG_CCSR_GPPORCR1)	32	R	<a href="#">See section</a>	13.3.3/488
1EE_0070	Device Disable Register 1 (DCFG_CCSR_DEVDISR1)	32	R/W	0000_0000h	13.3.4/488
1EE_0074	Device Disable Register 2 (DCFG_CCSR_DEVDISR2)	32	R/W	0000_0000h	13.3.5/490
1EE_0078	Device Disable Register 3 (DCFG_CCSR_DEVDISR3)	32	R/W	0000_0000h	13.3.6/492
1EE_007C	Device Disable Register 4 (DCFG_CCSR_DEVDISR4)	32	R/W	0000_0000h	13.3.7/493
1EE_0080	Device Disable Register 5 (DCFG_CCSR_DEVDISR5)	32	R/W	0000_0000h	13.3.8/494
1EE_0094	Core Disable Register (DCFG_CCSR_COREDISR)	32	R/W	0000_0000h	13.3.9/497
1EE_00A4	System Version Register (DCFG_CCSR_SVR)	32	R	<a href="#">See section</a>	13.3.10/499
1EE_00B0	Reset Control Register (DCFG_CCSR_RSTCR)	32	R/W	0000_0000h	13.3.11/500
1EE_00B4	Reset Request Preboot Loader Status Register (DCFG_CCSR_RSTRQPBLSR)	32	w1c	0000_0000h	13.3.12/501
1EE_00C0	Reset Request Mask Register (DCFG_CCSR_RSTRQMR1)	32	R/W	0000_4000h	13.3.13/502
1EE_00C8	Reset Request Status Register (DCFG_CCSR_RSTRQSR1)	32	w1c	0000_0000h	13.3.14/504
1EE_00E4	Boot Release Register (DCFG_CCSR_BRR)	32	R/W	0000_0000h	13.3.15/508
1EE_0100	Reset Control Word Status Register n (DCFG_CCSR_RCWSR1)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0104	Reset Control Word Status Register n (DCFG_CCSR_RCWSR2)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0108	Reset Control Word Status Register n (DCFG_CCSR_RCWSR3)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_010C	Reset Control Word Status Register n (DCFG_CCSR_RCWSR4)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0110	Reset Control Word Status Register n (DCFG_CCSR_RCWSR5)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0114	Reset Control Word Status Register n (DCFG_CCSR_RCWSR6)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0118	Reset Control Word Status Register n (DCFG_CCSR_RCWSR7)	32	R	<a href="#">See section</a>	13.3.16/509

*Table continues on the next page...*

## DCFG\_CCSR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1EE_011C	Reset Control Word Status Register n (DCFG_CCSR_RCWSR8)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0120	Reset Control Word Status Register n (DCFG_CCSR_RCWSR9)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0124	Reset Control Word Status Register n (DCFG_CCSR_RCWSR10)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0128	Reset Control Word Status Register n (DCFG_CCSR_RCWSR11)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_012C	Reset Control Word Status Register n (DCFG_CCSR_RCWSR12)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0130	Reset Control Word Status Register n (DCFG_CCSR_RCWSR13)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0134	Reset Control Word Status Register n (DCFG_CCSR_RCWSR14)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0138	Reset Control Word Status Register n (DCFG_CCSR_RCWSR15)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_013C	Reset Control Word Status Register n (DCFG_CCSR_RCWSR16)	32	R	<a href="#">See section</a>	13.3.16/509
1EE_0200	Scratch Read / Write Register n (DCFG_CCSR_SCRATCHRW1)	32	R/W	0000_0000h	<a href="#">13.3.17/510</a>
1EE_0204	Scratch Read / Write Register n (DCFG_CCSR_SCRATCHRW2)	32	R/W	0000_0000h	<a href="#">13.3.17/510</a>
1EE_0208	Scratch Read / Write Register n (DCFG_CCSR_SCRATCHRW3)	32	R/W	0000_0000h	<a href="#">13.3.17/510</a>
1EE_020C	Scratch Read / Write Register n (DCFG_CCSR_SCRATCHRW4)	32	R/W	0000_0000h	<a href="#">13.3.17/510</a>
1EE_0300	Scratch Read Register n (DCFG_CCSR_SCRATCHW1R1)	32	R/W	0000_0000h	<a href="#">13.3.18/511</a>
1EE_0304	Scratch Read Register n (DCFG_CCSR_SCRATCHW1R2)	32	R/W	0000_0000h	<a href="#">13.3.18/511</a>
1EE_0308	Scratch Read Register n (DCFG_CCSR_SCRATCHW1R3)	32	R/W	0000_0000h	<a href="#">13.3.18/511</a>
1EE_030C	Scratch Read Register n (DCFG_CCSR_SCRATCHW1R4)	32	R/W	0000_0000h	<a href="#">13.3.18/511</a>
1EE_0400	Core Reset Status Register n (DCFG_CCSR_CRSTS0)	32	R/W	0000_0000h	<a href="#">13.3.19/511</a>
1EE_0404	Core Reset Status Register n (DCFG_CCSR_CRSTS1)	32	R/W	0000_0000h	<a href="#">13.3.19/511</a>
1EE_0408	Core Reset Status Register n (DCFG_CCSR_CRSTS2)	32	R/W	0000_0000h	<a href="#">13.3.19/511</a>
1EE_040C	Core Reset Status Register n (DCFG_CCSR_CRSTS3)	32	R/W	0000_0000h	<a href="#">13.3.19/511</a>
1EE_0608	DMA Control Register (DCFG_CCSR_DMACR1)	32	R/W	0000_0000h	<a href="#">13.3.20/514</a>
1EE_0740	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP0)	32	R	<a href="#">See section</a>	<a href="#">13.3.21/515</a>
1EE_0744	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP1)	32	R	<a href="#">See section</a>	<a href="#">13.3.21/515</a>
1EE_0748	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP2)	32	R	<a href="#">See section</a>	<a href="#">13.3.21/515</a>
1EE_074C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP3)	32	R	<a href="#">See section</a>	<a href="#">13.3.21/515</a>

Table continues on the next page...

## DCFG\_CCSR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
1EE_0750	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP4)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0754	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP5)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0758	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP6)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_075C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP7)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0760	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP8)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0764	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP9)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0768	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP10)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_076C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP11)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0770	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP12)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0774	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP13)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0778	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP14)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_077C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP15)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0780	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP16)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0784	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP17)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0788	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP18)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_078C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP19)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0790	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP20)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0794	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP21)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0798	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP22)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_079C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP23)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07A0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP24)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07A4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP25)	32	R	<a href="#">See section</a>	13.3.21/515

Table continues on the next page...

## DCFG\_CCSR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1EE_07A8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP26)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07AC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP27)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07B0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP28)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07B4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP29)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07B8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP30)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07BC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP31)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07C0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP32)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07C4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP33)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07C8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP34)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07CC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP35)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07D0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP36)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07D4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP37)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07D8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP38)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07DC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP39)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07E0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP40)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07E4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP41)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07E8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP42)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07EC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP43)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07F0	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP44)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07F4	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP45)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07F8	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP46)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_07FC	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP47)	32	R	<a href="#">See section</a>	13.3.21/515

Table continues on the next page...

## DCFG\_CCSR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
1EE_0800	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP48)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0804	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP49)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0808	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP50)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_080C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP51)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0810	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP52)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0814	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP53)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0818	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP54)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_081C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP55)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0820	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP56)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0824	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP57)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0828	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP58)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_082C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP59)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0830	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP60)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0834	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP61)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0838	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP62)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_083C	Topology Initiator Type n Register (DCFG_CCSR_TP_ITYP63)	32	R	<a href="#">See section</a>	13.3.21/515
1EE_0844	Core Cluster n Topology Register (DCFG_CCSR_TP_CLUSTER1)	32	R	<a href="#">See section</a>	13.3.22/516
1EE_0E60	DDR Clock Disable Register (DCFG_CCSR_DDRCLKDR)	32	R/W	0000_0000h	13.3.23/517
1EE_0E68	IFC Clock Disable Register (DCFG_CCSR_IFCCLKDR)	32	R/W	0000_0000h	13.3.24/518
1EE_0E80	eSDHC Polarity Configuration Register (DCFG_CCSR_SDHCPCCR)	32	R/W	0000_0000h	13.3.25/519

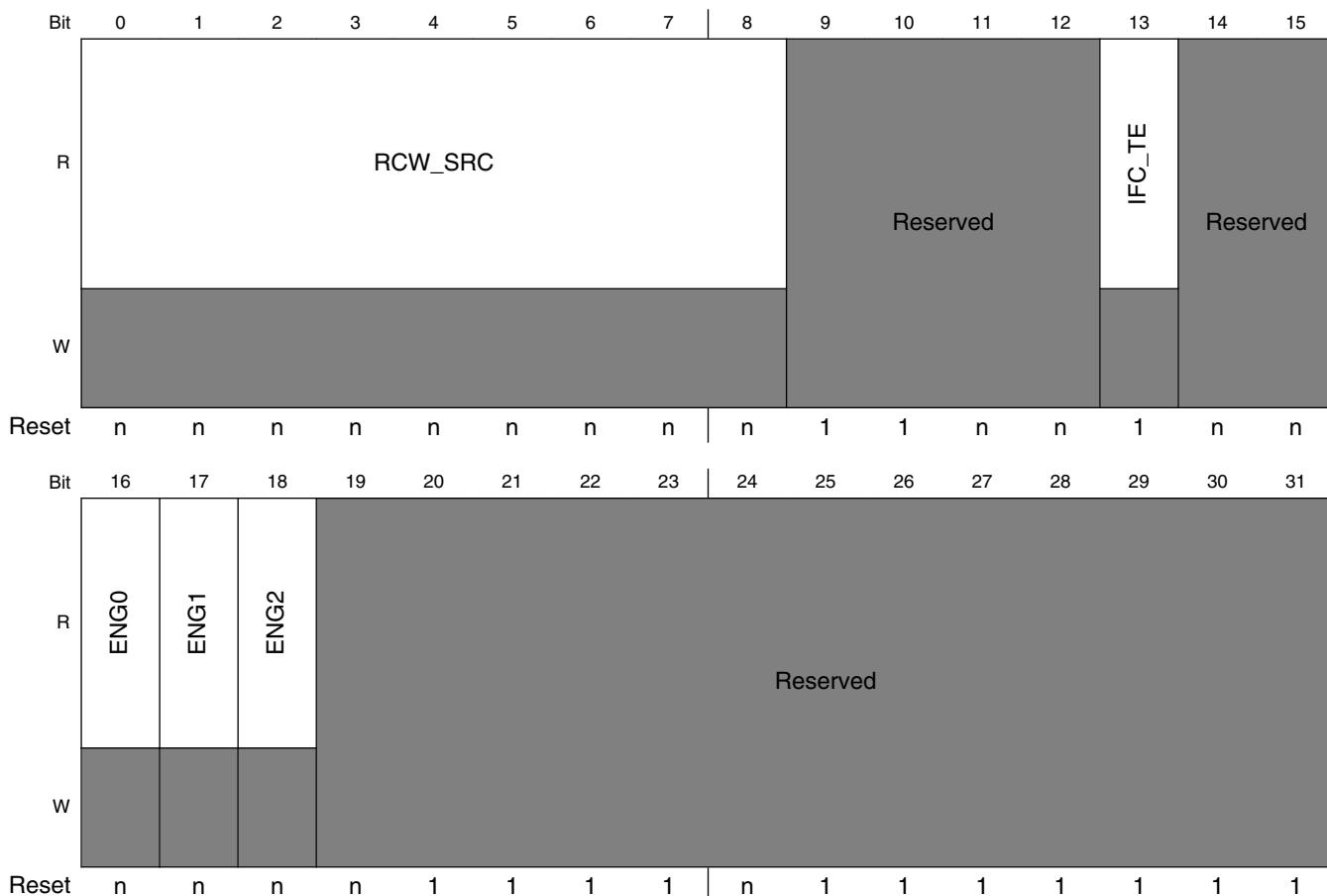
### 13.3.1 POR Status Register 1 (DCFG\_CCSR\_PORSR1)

PORSR1 captures the values of the device's POR configuration pins.

#### NOTE

The status of these bits depends on the sampling value driven on the pad.

Address: 1EE\_0000h base + 0h offset = 1EE\_0000h



**DCFG\_CCSR\_PORSR1 field descriptions**

Field	Description
0–8 RCW_SRC	Reset Configuration Word Source. Indicates which Flash memory contains the RCW information. Loaded with the value of cfg_rcw_src[0:8].
9–12 -	This field is reserved. Reserved
13 IFC_TE	IFC External Transceive Enable Polarity Selection. Loaded with the value of the cfg_ifc_te pin at power-on reset.

*Table continues on the next page...*

**DCFG\_CCSR\_PORSR1 field descriptions (continued)**

Field	Description
	0 Transceiver Enable polarity is configured active high 1 Transceiver Enable polarity is configured active low
14–15 -	This field is reserved. Reserved
16 ENG0	Engineering Use Loaded with the value of the cfg_eng_use[0] pin at power-on reset. 0 Config pin was not set 1 Config pin was set
17 ENG1	Engineering Use Loaded with the value of the cfg_eng_use[1] pin at power-on reset. 0 Config pin was not set 1 Config pin was set
18 ENG2	Engineering Use Loaded with the value of the cfg_eng_use[2] pin at power-on reset. 0 Config pin was not set 1 Config pin was set
19–31 -	This field is reserved. Reserved

**13.3.2 POR Status Register 2 (DCFG\_CCSR\_PORSR2)**

PORSR2 captures the values of the device's POR configuration pins.

**NOTE**

The status of these bits depends on the sampling value driven on the pad.

Address: 1EE\_0000h base + 4h offset = 1EE\_0004h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R			DRAM_TYPE													
	Reserved									Reserved						
W																
Reset	n	n	n	n	n	n	n	n	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCFG\_CCSR\_PORSR2 field descriptions**

Field	Description
0–1 -	This field is reserved. reserved
2 DRAM_TYPE	DRAM Type Selector. 0 DDR4 technology (1.2 V) 1 DDR3L technology (1.35 V)
3–31 -	This field is reserved. Reserved

### 13.3.3 General-Purpose POR Configuration Register (DCFG\_CCSR\_GPPORCR1)

GPPORCR1 stores the value sampled from the integrated Flash controller address/data signals, IFC\_AD[0:7], using sampling logic similar to that for the configuration pins. It is provided for customer use and always samples 8-bits, independent of the width of the chip's flash controller's LAD bus. Software can use this value to inform the operating system about initial system configuration. Typical interpretations include circuit board type, board ID number, or a list of available peripherals.

Address: 1EE\_0000h base + 20h offset = 1EE\_0020h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	POR_CFG_VEC								Reserved																							
W																																
Reset	*	*	*	*	*	*	*	*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

\* Notes:

- POR\_CFG\_VEC field: Reset value supplied by cfg\_gpininput[0:7].

#### DCFG\_CCSR\_GPPORCR1 field descriptions

Field	Description
0–7 POR_CFG_VEC	General-purpose POR configuration vector sampled from integrated Flash controller address/data signals, IFC_AD[0:7] signals (uppermost 8-bits) at the negation of PORESET.  <b>NOTE:</b> Reset value supplied by cfg_gpininput[0:7].
8–31 -	This field is reserved. Reserved

### 13.3.4 Device Disable Register 1 (DCFG\_CCSR\_DEVDISR1)

A given application may not use all the peripherals on the device. In this case, it may be desirable to disable unused peripherals. DEVDISR1 provides a mechanism for gating clocks to IP blocks that are not used when running an application.

#### NOTE

IP blocks disabled by setting the corresponding bit in the DEVDISR1 register must not be re-enabled.

#### NOTE

Power down for any module can be configured by programming the DCFG\_DEVDISR register. If any module is

powered down, then its data access as well as register access are not allowed.

Address: 1EE\_0000h base + 70h offset = 1EE\_0070h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
	PBL	Reserved	ESDHC						DMA1	DMA2					USB3	USB2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
	SATA	USB1		Reserved		SEC									Reserved	QE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCFG\_CCSR\_DEVDIR1 field descriptions

Field	Description
0 PBL	Pre-boot loader disable. 0 Module is enabled 1 Module is disabled
1 -	This field is reserved. Reserved
2 ESDHC	eSDHC controller disable. 0 Module is enabled 1 Module is disabled
3–7 -	This field is reserved. Reserved
8 DMA1	DMA controller 1 disable. DMA1 here represents qDMA. 0 Module is enabled 1 Module is disabled
9 DMA2	DMA controller 2 disable. DMA2 here represents eDMA.

Table continues on the next page...

**DCFG\_CCSR\_DEVDISR1 field descriptions (continued)**

Field	Description
	0 Module is enabled 1 Module is disabled
10–13 -	This field is reserved. Reserved
14 USB3	USB controller 3 disable. 0 Module is enabled 1 Module is disabled
15 USB2	USB Controller 2 disable. 0 Module is enabled 1 Module is disabled
16 SATA	SATA disable. 0 Module is enabled 1 Module is disabled
17 USB1	USB controller 1 disable. 0 Module is enabled 1 Module is disabled
18–21 -	This field is reserved. Reserved
22 SEC	SEC module disable. 0 Module is enabled 1 Module is disabled
23–29 -	This field is reserved. Reserved
30 -	This field is reserved.
31 QE	QUICC Engine disable. 0 Module is enabled 1 Module is disabled

**13.3.5 Device Disable Register 2 (DCFG\_CCSR\_DEVDISR2)**

A given application may not use all the peripherals on the device. In this case, it may be desirable to disable unused peripherals. DEVDISR2 provides a mechanism for gating clocks to IP blocks that are not used when running an application.

**NOTE**

IP Blocks disabled by setting the corresponding bit in the DEVDISR2 register must not be re-enabled.

**NOTE**

Power down for any module can be configured by programming the DCFG\_DEVDISR register. If any module is powered down, then its data access as well as register access are not allowed.

Address: 1EE\_0000h base + 74h offset = 1EE\_0074h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	FMAN1_MAC1	FMAN1_MAC2	FMAN1_MAC3	FMAN1_MAC4	FMAN1_MAC5	FMAN1_MAC6	Reserved		FMAN1_MAC9	Reserved						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W									FMAN1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCFG\_CCSR\_DEVDISR2 field descriptions**

Field	Description
0 FMAN1_MAC1	Frame manager 1, MAC1 disable. 0 Module is enabled 1 Module is disabled
1 FMAN1_MAC2	Frame manager 1, MAC2 disable. 0 Module is enabled 1 Module is disabled
2 FMAN1_MAC3	Frame manager 1, MAC3 disable. 0 Module is enabled 1 Module is disabled
3 FMAN1_MAC4	Frame manager 1, MAC4 disable. 0 Module is enabled 1 Module is disabled

Table continues on the next page...

**DCFG\_CCSR\_DEVDISR2 field descriptions (continued)**

Field	Description
4 FMAN1_MAC5	Frame manager 1, MAC5 disable. 0 Module is enabled 1 Module is disabled
5 FMAN1_MAC6	Frame manager 1, MAC6 disable. 0 Module is enabled 1 Module is disabled
6–7 -	This field is reserved. Reserved
8 FMAN1_MAC9	Frame manager 1, MAC9 disable. 0 Module is enabled 1 Module is disabled
9 -	This field is reserved. Reserved
10–23 -	This field is reserved. Reserved
24 FMAN1	Frame manager 1 disable. 0 Module is enabled 1 Module is disabled
25–31 -	This field is reserved. Reserved

**13.3.6 Device Disable Register 3 (DCFG\_CCSR\_DEVDISR3)**

A given application may not use all the peripherals on the device. In this case, it may be desirable to disable unused peripherals. DEVDISR3 provides a mechanism for gating clocks to IP blocks that are not used when running an application.

**NOTE**

IP Blocks disabled by setting the corresponding bit in the DEVDISR3 register must not be re-enabled.

**NOTE**

Power down for any module can be configured by programming the DCFG\_DEVDISR register. If any module is powered down, then its data access as well as register access are not allowed.

Address: 1EE\_0000h base + 78h offset = 1EE\_0078h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved				QMAN	BMAN		Reserved
W													0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCFG\_CCSR\_DEVDISR3 field descriptions

Field	Description
0–11 -	This field is reserved. Reserved
12 QMAN	Queue manager disable. 0 Module is enabled 1 Module is disabled
13 BMAN	Buffer manager disable. 0 Module is enabled 1 Module is disabled
14–31 -	This field is reserved. Reserved

### 13.3.7 Device Disable Register 4 (DCFG\_CCSR\_DEVDISR4)

A given application may not use all the peripherals on the device. In this case, it may be desirable to disable unused peripherals. DEVDISR4 provides a mechanism for gating clocks to IP blocks that are not used when running an application.

#### NOTE

IP Blocks disabled by setting the corresponding bit in the DEVDISR4 register must not be re-enabled.

#### NOTE

Power down for any module can be configured by programming the DCFG\_DEVDISR register. If any module is powered down, then its data access as well as register access are not allowed.

## Device Configuration/Pin Control Memory Map

Address: 1EE\_0000h base + 7Ch offset = 1EE\_007Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	DUART1	DUART2	QSPI												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCFG\_CCSR\_DEVDISR4 field descriptions

Field	Description
0–1 -	This field is reserved. Reserved
2 DUART1	DUART1 module disable. 0 Module is enabled 1 Module is disabled
3 DUART2	DUART2 module disable. 0 Module is enabled 1 Module is disabled
4 QSPI	QuadSPI module disable 0 Module is enabled 1 Module is disabled
5–31 -	This field is reserved. Reserved

### 13.3.8 Device Disable Register 5 (DCFG\_CCSR\_DEVDISR5)

A given application may not use all the peripherals on the device. In this case, it may be desirable to disable unused peripherals. DEVDISR5 provides a mechanism for gating clocks to IP blocks that are not used when running an application.

#### NOTE

IP Blocks disabled by setting the corresponding bit in the DEVDISR5 register must not be re-enabled.

#### NOTE

Power down for any module can be configured by programming the DCFG\_DEVDISR register. If any module is

powered down, then its data access as well as register access are not allowed.

Address: 1EE\_0000h base + 80h offset = 1EE\_0080h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W	DDR	Reserved	LPUART4	Reserved	OCRAM1	OCRAM2			IFC	GPIO	DBG	Reserved	Reserved	LPUART1	LPUART2	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	LPUART3	Reserved	LPUART5	LPUART6	WDOG1	FlexTimer	WDOG2	SPI1	WDOG3	WDOG4	WDOG5	IIC4	IIC3	IIC2	IIC1	ICMMU
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCFG\_CCSR\_DEVDISR5 field descriptions

Field	Description
0 DDR	DDR Controller disable 0 Module is enabled 1 Module is disabled
1–2 -	This field is reserved. Reserved
3 LPUART4	Module disable 0 Module is enabled 1 Module is disabled
4–5 -	This field is reserved. Reserved
6 OCRAM1	OCRAM1 disable 0 Module is enabled 1 Module is disabled
7 OCRAM2	OCRAM2 disable 0 Module is enabled 1 Module is disabled

Table continues on the next page...

**DCFG\_CCSR\_DEVDISR5 field descriptions (continued)**

Field	Description
8 IFC	Integrated Flash controller disable.  0 Module is enabled 1 Module is disabled
9 GPIO	GPIO disable.  <b>NOTE:</b> This field disables all GPIO modules.  0 Module is enabled 1 Module is disabled
10 DBG	Debug module disable.  0 Module is enabled 1 Module is disabled
11 -	This field is reserved. Reserved
12–13 -	This field is reserved. Reserved
14 LPUART1	LPUART1 disable  0 Module is enabled 1 Module is disabled
15 LPUART2	LPUART2 disable  0 Module is enabled 1 Module is disabled
16 LPUART3	LPUART3 disable  0 Module is enabled 1 Module is disabled
17 -	This field is reserved. Reserved
18 LPUART5	LPUART5 disable  0 Module is enabled 1 Module is disabled
19 LPUART6	LPUART6 disable  0 Module is enabled 1 Module is disabled
20 WDOG1	WDOG1 disable.  0 Module is enabled 1 Module is disabled
21 FlexTimer	FlexTimer disable.  0 Module is enabled 1 Module is disabled

*Table continues on the next page...*

**DCFG\_CCSR\_DEVDISR5 field descriptions (continued)**

Field	Description
22 WDOG2	WDOG2 disable. 0 Module is enabled 1 Module is disabled
23 SPI1	SPI1 disable. 0 Module is enabled 1 Module is disabled
24 WDOG3	WDOG3 disable. 0 Module is enabled 1 Module is disabled
25 WDOG4	WDOG4 disable. 0 Module is enabled 1 Module is disabled
26 WDOG5	WDOG5 disable. 0 Module is enabled 1 Module is disabled
27 IIC4	IIC4 disable. 0 Module is enabled 1 Module is disabled
28 IIC3	IIC3 disable. 0 Module is enabled 1 Module is disabled
29 IIC2	IIC2 disable. 0 Module is enabled 1 Module is disabled
30 IIC1	IIC1 disable. 0 Module is enabled 1 Module is disabled
31 ICMMU	Interconnects and MMU disable. 0 Module is enabled 1 Module is disabled

**13.3.9 Core Disable Register (DCFG\_CCSR\_COREDISR)**

COREDISR provides a mechanism for gating clocks to any cores on the device that are not used when running an application.

COREDISR register should only be configured in the following conditions:

- Before system ready, COREDISR register can be programmed by the external debugger or Pre-Boot Initialization.
- After system ready, a COREDISR register bit can be programmed for the corresponding core by the external debugger or embedded software while the core is in boot-holdoff mode.

### NOTE

Cores that have an interrupt pending will not be disabled by COREDISR until the interrupt is cleared.

### NOTE

Cores disabled by setting the corresponding bit in the COREDISR register must not be re-enabled via software. The only supported mechanism for re-enabling a Core disabled in this manner is via power-on, hard reset, or core reset.

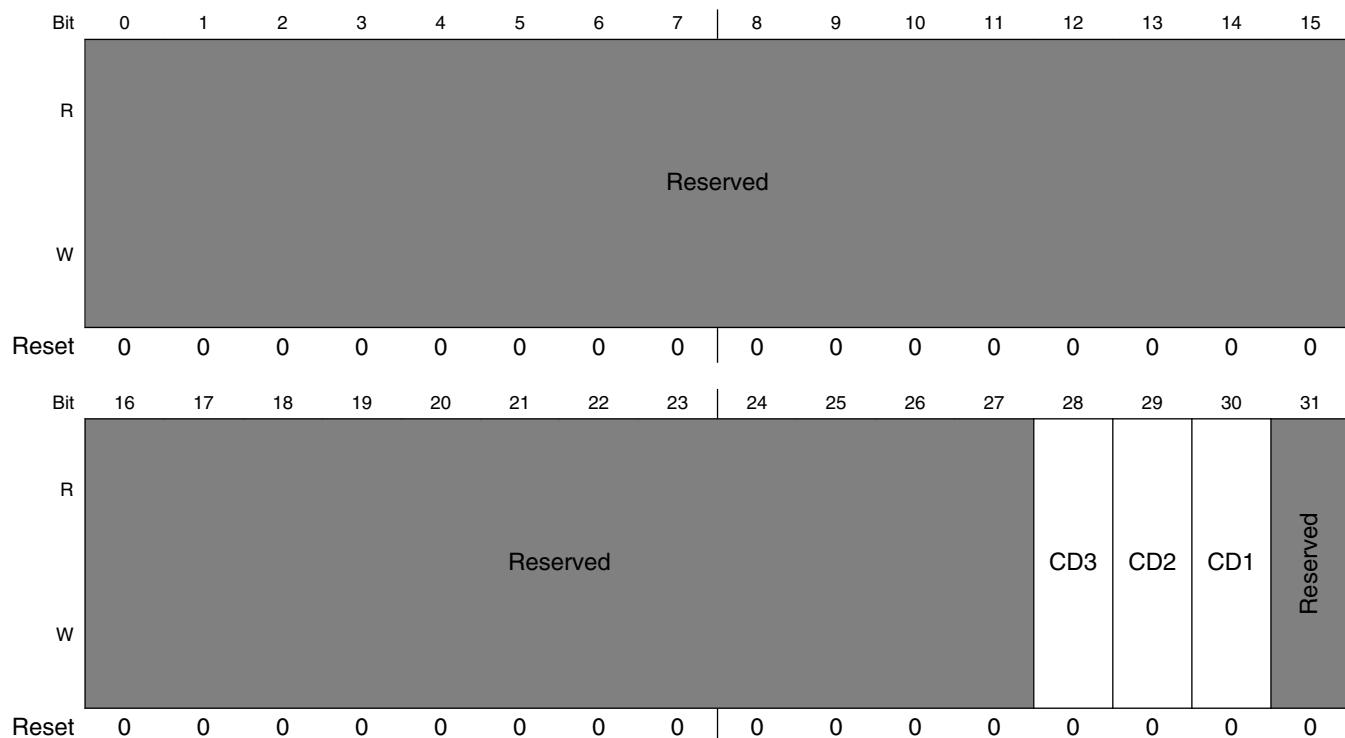
### NOTE

The LSB, bit 31, is associated with Core 0.

Note that

- for dual core personality, COREDISR will be set by hardware
- If software sets COREDISR to disable the cores, it will not work as halting core involves a software step (wfi execution)

Address: 1EE\_0000h base + 94h offset = 1EE\_0094h



**DCFG\_CCSR\_COREDISR field descriptions**

Field	Description
0–27 -	This field is reserved. Reserved
28 CD3	Core 3 Disable. 0 Selected Core is enabled 1 Selected Core is disabled
29 CD2	Core 2 Disable. 0 Selected Core is enabled 1 Selected Core is disabled
30 CD1	Core 1 Disable. 0 Selected Core is enabled 1 Selected Core is disabled
31 -	This field is reserved. Reserved, set to 0

**13.3.10 System Version Register (DCFG\_CCSR\_SVR)**

The SVR contains the system version number for the device. This value can also be read through the SVR SPR of the Arm Cortex-A53 core.

Address: 1EE\_0000h base + A4h offset = 1EE\_00A4h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	MFR_ID		SOC_DEV_ID														VAR_PER		MAJOR_REV		MINOR_REV											
W																																
Reset	1	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	0	0	0	n	0	0	n	0	0	0	1	0	0	0	1	

**DCFG\_CCSR\_SVR field descriptions**

Field	Description
0–3 MFR_ID	Manufacturer ID
4–15 SOC_DEV_ID	Chip Device ID
16–23 VAR_PER	Various Personalities For 21x21 package: 0000_0000 LS1043A (Export controlled crypto hardware enabled) 0000_0001 LS1043A (Export controlled crypto hardware disabled) 0000_1000 LS1023A (Export controlled crypto hardware enabled) 0000_1001 LS1023A (Export controlled crypto hardware disabled)

Table continues on the next page...

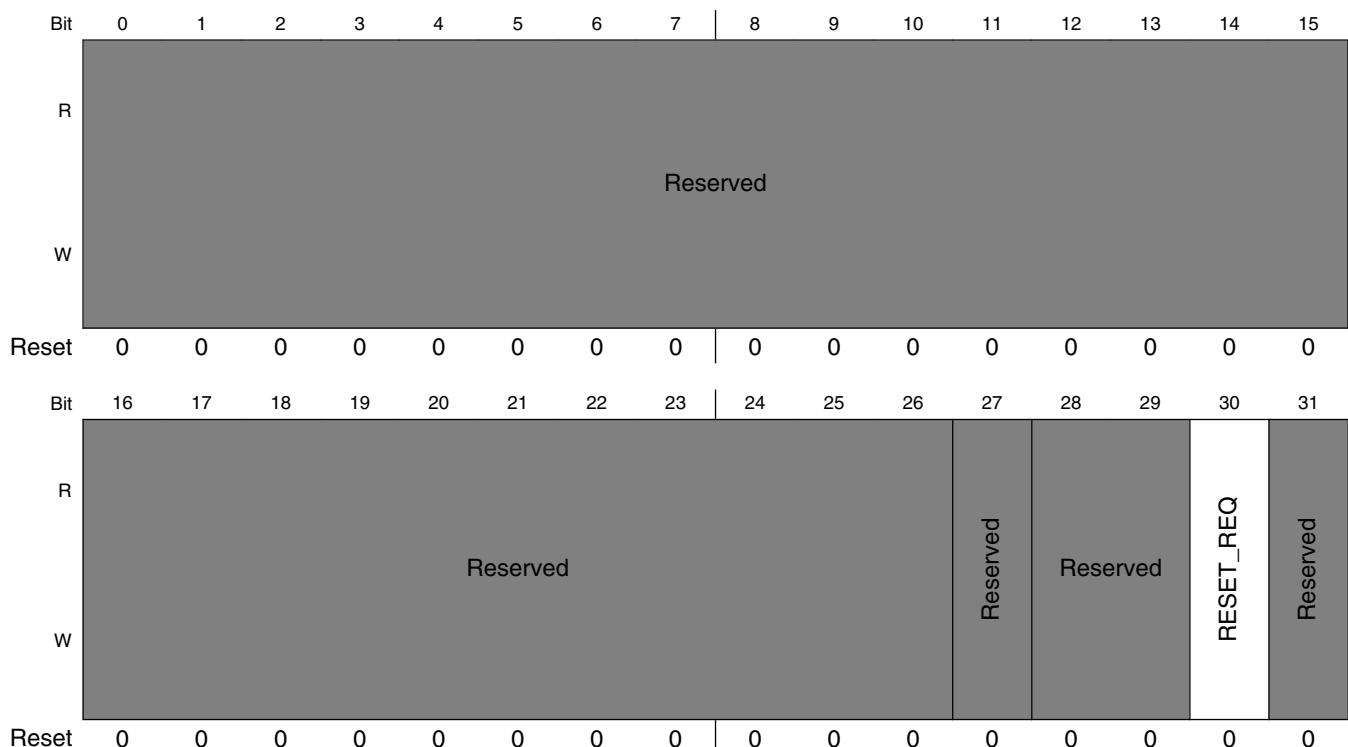
**DCFG\_CCSR\_SVR field descriptions (continued)**

Field	Description
24–27 MAJOR_REV	Major Revision Number
28–31 MINOR_REV	Minor Revision Number For Si 1.1, the minor revision is 0x1. For Si 1.0, the minor revision is 0x0.

**13.3.11 Reset Control Register (DCFG\_CCSR\_RSTCR)**

The RSTCR allows software to control reset functions.

Address: 1EE\_0000h base + B0h offset = 1EE\_00B0h

**DCFG\_CCSR\_RSTCR field descriptions**

Field	Description
0–26 -	This field is reserved. Reserved
27 -	This field is reserved. Reserved

*Table continues on the next page...*

**DCFG\_CCSR\_RSTCR field descriptions (continued)**

Field	Description
28–29 -	This field is reserved. Reserved
30 RESET_REQ	Hardware reset request. External hardware may then decide to issue the desired reset signal (PORESET_B or HRESET_B) to the device.  0 No reset request initiated. 1 Hardware reset request initiated by software.
31 -	This field is reserved. Reserved

**13.3.12 Reset Request Preboot Loader Status Register (DCFG\_CCSR\_RSTRQPBLSR)**

The RSTRQPBLSR contains status bits to record the reasons for RESET\_REQ\_B assertion. It excludes core watchdog timer sources.

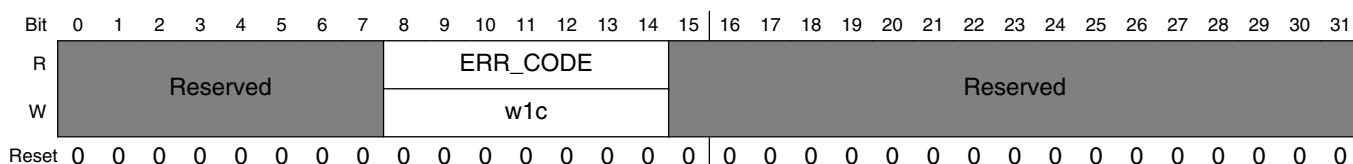
**NOTE**

This register's code is valid only when RSTRQSR[PBL\_RR] is set.

**NOTE**

ECC errors detected during NAND flash loading during RCW and PBI phases are reported in RSTRQSR[IFC\_RR]. See the description of this bit for more details.

Address: 1EE\_0000h base + B4h offset = 1EE\_00B4h

**DCFG\_CCSR\_RSTRQPBLSR field descriptions**

Field	Description
0–7 -	This field is reserved. Reserved
8–14 ERR_CODE	7-bit PBL Error Code  7-bit encoded value reflects one of 128 possible PBL errors.  Write 1 to each bit to clear this field.  <b>NOTE:</b> See <a href="#">Error codes</a> for details on the PBL error encodings.
15–31 -	This field is reserved. Reserved

### 13.3.13 Reset Request Mask Register (DCFG\_CCSR\_RSTQMR1)

The RSTRQMR contains mask bits for optional masking of RESET\_REQ\_B sources to prevent generation of such a reset request. It excludes core watchdog timer sources.

Address: 1EE\_0000h base + C0h offset = 1EE\_00C0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CORE_WDOG3_RST_MSK	CORE_WDOG4_RST_MSK	CORE_WDOG5_RST_MSK	Reserved	Reserved						IFC_RST_MSK	Reserved	ALTCBAR_RST_MSK	PBL_RST_MSK	SFP_RST_MSK	SEC_RST_MSK
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SDC_RST_MSK	MBEE_RST_MSK	Reserved	RPTOE_RST_MSK	SRDS_RST_MSK	CCP_ERR_RST_MSK	CORE_WDOG_RST_MSK	Reserved								
W	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCFG\_CCSR\_RSTQMR1 field descriptions**

Field	Description
0 CORE_WDOG3_RST_MSK	Core watchdog reset request mask. 0 Core watchdog reset request can cause a reset request 1 Core watchdog reset request cannot cause a reset request
1 CORE_WDOG4_RST_MSK	Core watchdog reset request mask. 0 Core watchdog reset request can cause a reset request 1 Core watchdog reset request cannot cause a reset request
2 CORE_WDOG5_RST_MSK	Core watchdog reset request mask. 0 Core watchdog reset request can cause a reset request 1 Core watchdog reset request cannot cause a reset request
3 -	This field is reserved. Reserved

*Table continues on the next page...*

**DCFG\_CCSR\_RSTRQMR1 field descriptions (continued)**

Field	Description
4–8 -	This field is reserved. Reserved
9 IFC_MSK	Integrated Flash Controller reset request event mask 0 IFC error event can cause a reset request 1 IFC error event cannot cause a reset request
10–11 -	This field is reserved. Reserved
12 ALTCBAR_MSK	ALTCBAR violation by PBL reset request mask 0 ALTCBAR violation by PBL can cause a reset request 1 ALTCBAR violation by PBL cannot cause a reset request
13 PBL_MSK	PBL error reset request event mask. 0 PBL error event can cause a reset request 1 PBL error event cannot cause a reset request
14 SFP_MSK	Security Fuse Processor error during POR fuse process reset mask. 0 Security Fuse Processor error event can cause a reset request 1 Security Fuse Processor error event cannot cause a reset request
15 SEC_MSK	Security monitor reset request event mask 0 Security monitor reset request event mask can cause a reset request 1 Security monitor reset request event mask cannot cause a reset request
16 SDC_MSK	Security Debug Controller error reset request mask 0 SDC error event can cause a reset request 1 SDC error event cannot cause a reset request
17 MBEE_MSK	Multi-bit ECC error reset request mask 0 Multi-bit ECC error event can cause a reset request 1 Multi-bit ECC error event cannot cause a reset request
18 -	This field is reserved. Reserved  POR BIST error event can cause a reset request
19 RPTOE_MSK	RCPM Time Out reset request event mask 0 RCPM Time Out event can cause a reset request 1 RCPM Time Out event cannot cause a reset request
20 SRDS_RST_MSK	SerDes reset request event mask 0 SerDes reset event can cause a reset request 1 SerDes reset event cannot cause a reset request
21 CCP_ERR_MSK	REP Error Alarm from CCP(REP) event mask. 0 REP Error Alarm from CCP event can cause a reset request 1 REP Error Alarm from CCP event cannot cause a reset request

*Table continues on the next page...*

**DCFG\_CCSR\_RSTRQMR1 field descriptions (continued)**

Field	Description
22 CORE_WDOG_ RST_MSK	Core watchdog reset request mask.  0 Core watchdog reset request can cause a reset request 1 Core watchdog reset request cannot cause a reset request
23–31 -	This field is reserved. Reserved

**13.3.14 Reset Request Status Register (DCFG\_CCSR\_RSTRQSR1)**

The RSTRQSR contains status bits to record the reasons for RESET\_REQ\_B assertion. The bits here are set independent of the RSTRQMR. This means if a reset reason occurs and is masked by RSTRQMR, it will still be recorded in RSTRQSR.

**NOTE**

For the different sources captured in this register, some of these can be serviced with either PORESET\_B or HRESET\_B and some of these must be serviced with PORESET\_B only.

Address: 1EE\_0000h base + C8h offset = 1EE\_00C8h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CORE_WDOG3_RST_RR	CORE_WDOG4_RST_RR	CORE_WDOG5_RST_RR	Reserved	Reserved					IFC_RR	Reserved		ALTCBAR_RR	PBL_RR	SFP_RR	SEC_RR
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## Device Configuration/Pin Control Memory Map

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SDC_RR	MBEE_RR	Reserved	RPTOE_RR	SRDS_RST_RR	CCP_ERR_RR	CORE_WDOG1_RST_RR									
W														Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCFG\_CCSR\_RSTRQSR1 field descriptions

Field	Description
0 CORE_WDOG3_RST_RR	Core watchdog event reset request 0 WDOG reset request from WDOG3 is not active 1 WDOG reset request from WDOG3 is active
1 CORE_WDOG4_RST_RR	Core watchdog event reset request 0 WDOG reset request from WDOG4 is not active 1 WDOG reset request from WDOG4 is active
2 CORE_WDOG5_RST_RR	Core watchdog event reset request 0 WDOG reset request from WDOG5 is not active 1 WDOG reset request from WDOG5 is active
3 -	This field is reserved. Reserved
4–8 -	This field is reserved. Reserved
9 IFC_RR	Integrated Flash Controller reset request event

Table continues on the next page...

**DCFG\_CCSR\_RSTRQSR1 field descriptions (continued)**

Field	Description
	<p>This bit is set if an ECC error occurred in NAND Flash preload for the RCW phase or the Preboot Initialization phase</p> <p><b>NOTE:</b> After a PORESET, RCWSRn registers can be read. If RSTRQSR[IFC_RR] is set after a PORESET and RCWSRn does not contain RCW values, then the failure occurred during RCW. If the bit is set and RCWSRn contains valid values, then the failure occurred during PBI.</p> <p>0 IFC reset event reset request event not active 1 IFC reset event reset request event active</p>
10–11 -	This field is reserved. Reserved
12 ALTCBAR_RR	<p>ALTCBAR violation by PBL reset request</p> <p>0 ALTCBAR violation by PBL reset request event not active 1 ALTCBAR violation by PBL reset request event active</p>
13 PBL_RR	<p>PBL error reset request requires device level PORESET_B or HRESET_B.</p> <p>0 PBL reset request event not active 1 PBL reset request event active</p>
14 SFP_RR	<p>Security Fuse Processor error during POR fuse process caused reset request.</p> <p>Security Fuse Processor error requires device level PORESET_B</p> <p>0 Security Fuse Processor reset request event not active 1 Security Fuse Processor reset request event active</p>
15 SEC_RR	<p>Security Monitor detected security violation/tampering event during POR fuse process caused reset request.</p> <p>Security Monitor reached Hard Fail state and requires device level PORESET_B .</p> <p>0 Security Monitor reset request event not active 1 Security Monitor reset request event active</p>
16 SDC_RR	<p>Security Debug Controller reset request</p> <p>Security Debug Controller requires device level PORESET_B .</p> <p>0 SDC reset request event not active 1 SDC reset request event active</p>
17 MBEE_RR	<p>Multi-bit ECC reset request</p> <p>Platform internal memory multi-bit ECC error requires device level PORESET_B or HRESET_B.</p> <p><b>Note:</b> This bit is for multi-bit error in any of the SRAMs inside the chip including OCRAM and not for L1/L2 cache memories.</p> <p>0 Multi-bit ECC error reset request event not active 1 Multi-bit ECC error reset request event active</p>
18 -	This field is reserved. Reserved
19 RPTOE_RR	<p>RCPM Time Out reset request event</p> <p>RCPM Time Out event (for core halt, core stop, or core reset request) requires device level PORESET_B or HRESET_B.</p>

*Table continues on the next page...*

**DCFG\_CCSR\_RSTRQSR1 field descriptions (continued)**

Field	Description
	0 RCPM Time Out event reset request event not active 1 RCPM Time Out event reset request event active
20 SRDS_RST_RR	SerDes reset event. Occurs if any enabled SerDes PLL does not lock. 0 SerDes reset request event not active 1 SerDes reset request event active
21 CCP_ERR_RR	REP Error Alarm from CCP(REP) event reset request.
22 CORE_WDOG1_RST_RR	Core watchdog event reset request. 0 Core watchdog reset request from WDOG1 is not active 1 Core watchdog reset request from WDOG1 is active
23–31 -	This field is reserved. Reserved

**13.3.15 Boot Release Register (DCFG\_CCSR\_BRR)**

The BRR contains control bits for enabling boot for each core. On exiting HRESET or PORESET, the RCW BOOT\_HO field optionally allows for logical core 0 to be released for booting or to remain in boot holdoff. All other cores remain in boot holdoff until their corresponding bit is set.

This is configured by the Internal boot ROM (IBR) code based on the [Core Disable Register \(DCFG\\_CCSR\\_COREDISR\)](#).

**NOTE**

The LSB, bit 31, is associated with Core 0.

**NOTE**

If a bit is changed from 1 to 0 outside of warm reset (at runtime), results are boundedly undefined for that core.

Address: 1EE\_0000h base + E4h offset = 1EE\_00E4h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	Reserved															
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	CR3 CR2 CR1 CR0															

## **DCFG\_CCSR\_BRR field descriptions**

Field	Description
0–27 -	This field is reserved. Reserved
28 CR3	Core 3 Release.  0 Core is in Boot Holdoff and not released for Booting 1 Core released for Booting
29 CR2	Core 2 Release.  0 Core is in Boot Holdoff and not released for Booting 1 Core released for Booting
30 CR1	CR1  Core 1 Release.
	0 Core is in Boot Holdoff and not released for Booting 1 Core released for Booting
31 CR0	CR0  Core 0 Release.
	0 Core is in Boot Holdoff and not released for Booting 1 Core released for Booting

### **13.3.16 Reset Control Word Status Register n (DCFG\_CCSR\_RCWSR $n$ )**

RCWSR contains the Reset Configuration Word (RCW) information written with values read from flash memory by the device at power-on reset and read-only upon exiting reset.

## NOTE

After a PORESET, RCWSRn registers can be read. If RSTRQSR[IFC\_RR] is set after a PORESET and RCWSRn does not contain RCW values, then the failure occurred during RCW. If the bit is set and RCWSRn contains valid values, then the failure occurred during PBI.

Address: 1EE 0000h base + 100h offset + (4d × i), where i=0d to 15d

**DCFG\_CCSR\_RCWSR $n$  field descriptions**

Field	Description
0-31 RCW	Read-only value of RCW bits $(n-1)*32 : (n*32)-1$ loaded in power-on reset's Reset Configuration stage.

### 13.3.17 Scratch Read / Write Register n (DCFG\_CCSR\_SCRATCHRW $n$ )

The SCRATCHRW $n$  provides read / write scratch register locations available to the user.

**NOTE**

When performing secure boot, these registers are defined as follows:

- SCRATCHRW1 - Pointer to ESBC Header (Primary Boot Image)
- SCRATCHRW2 - Failure Code if Secure Boot Fails (Primary Boot Image)
- SCRATCHRW3 - Pointer to ESBC Header (Alternate Boot Image)
- SCRATCHRW4 - Failure Code if Secure Boot Fails (Alternate Boot Image)

Address: 1EE\_0000h base + 200h offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCFG\_CCSR\_SCRATCHRW $n$  field descriptions**

Field	Description
0-31 VAL	32-bit scratch contents

### 13.3.18 Scratch Read Register n (DCFG\_CCSR\_SCRATCHW1Rn)

The SCRATCHW1Rn provides scratch register locations available to the user. These are write-once registers. After these have been written once, they can only be written after a power-on or hard reset.

Address: 1EE\_0000h base + 300h offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### DCFG\_CCSR\_SCRATCHW1Rn field descriptions

Field	Description
0–31 VAL	32-bit scratch contents

### 13.3.19 Core Reset Status Register n (DCFG\_CCSR\_CRSTSFn)

The CRSTSFn contains the reset status bits for each thread on the device.

In this register (one per physical core):

- Bits 0-23 reflect per-thread conditions for this specific core
- Bits 24-31 reflect device conditions which in turn affect this specific core

#### NOTE

The number of CRSTSFn registers is determined by the number of physical cores.

#### Reset of this Register

A power-on reset of the device causes the following to occur:

- RST\_PORST is set
- All other bits are cleared

A hard reset of the device causes the following to occur:

- RST\_PORST remains unchanged (PORST resources are not affected by Hard Reset)
- RST\_HRST is set
- All other bits are cleared

## Device Configuration/Pin Control Memory Map

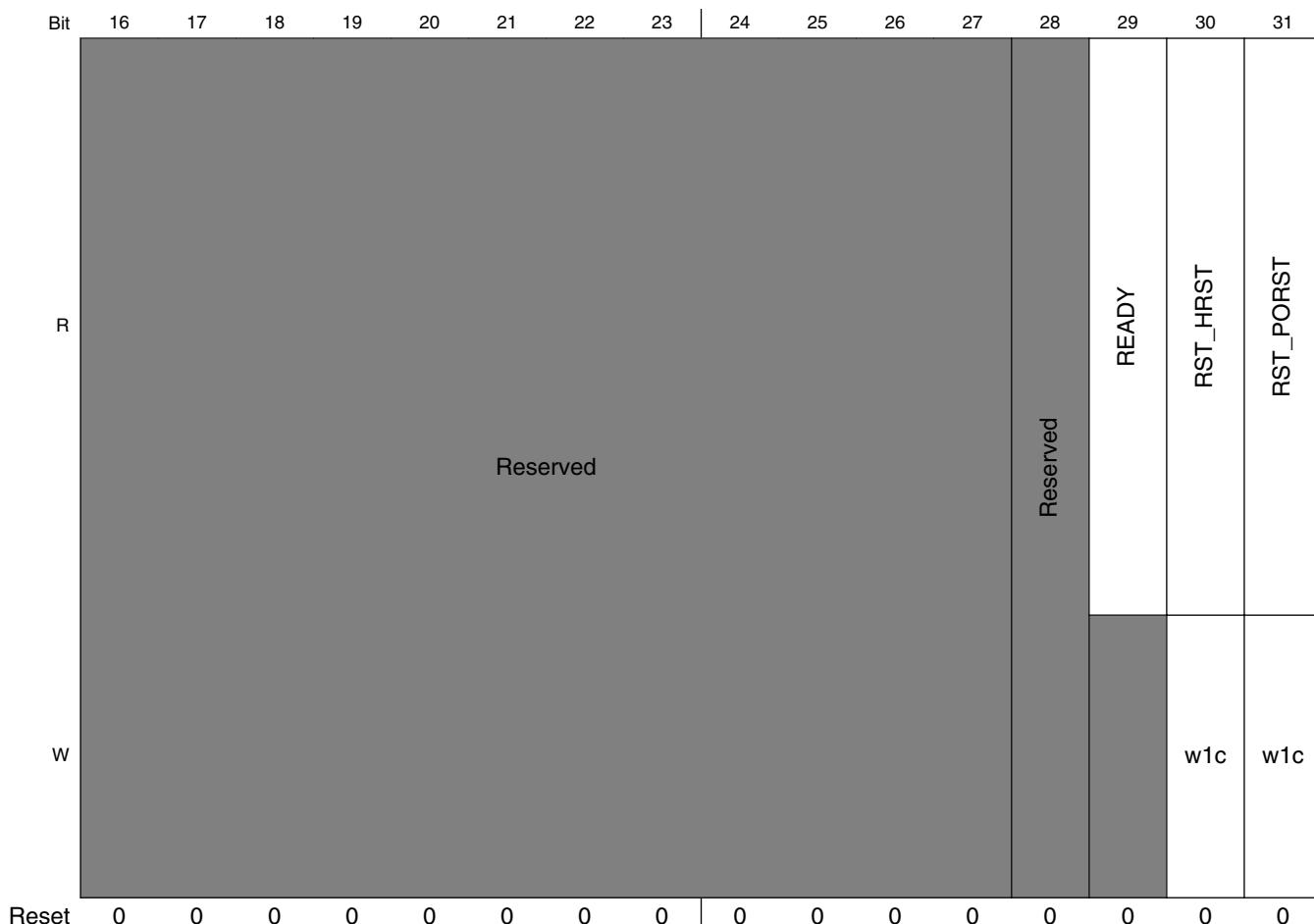
If an application prefers that RST\_PORST is not set after hard reset, then a write-1-clear must be done to CRSTS<sub>Rn</sub>[RST\_PORST] upon exiting power-on reset.

### Ready Bit Functionality

This bit is cleared on a device power-on or hard reset. Upon completion of power-on or hard reset processing, this bit may be automatically set for a core if none of the conditions specified in the READY bit definition are true.

Address: 1EE\_0000h base + 400h offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



### DCFG\_CCSR\_CRSTSR $n$ field descriptions

Field	Description
0–5 -	This field is reserved. Reserved
6–7 -	This field is reserved. Reserved
8–13 -	This field is reserved. Reserved
14–15 -	This field is reserved. Reserved
16–27 -	This field is reserved.
28 -	This field is reserved. Reserved
29 READY	Core ready pin. Core is in the 'ready' state after device has successfully passed through the "System Ready" point and the core is currently not in any of the following states:  This bit reflects what is driven on the READY_P0 external signal.

Table continues on the next page...

**DCFG\_CCSR\_CRSTS $n$  field descriptions (continued)**

Field	Description
	0 Core 0 not ready 1 Core 0 ready
30 RST_HRST	Core was reset due to an HRESET (note a PORESET causes an HRESET, but a PORESET will not cause this bit to be set)
31 RST_PORST	Core was reset due to a PORESET

**13.3.20 DMA Control Register (DCFG\_CCSR\_DMACR1)**

The DMACR1 contains bits for allowing DMA transactions (qDMA) from internal sources on the device.

Address: 1EE\_0000h base + 608h offset = 1EE\_0608h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	DMA1_0	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCFG\_CCSR\_DMCR1 field descriptions**

Field	Description
0–1 DMA1_0	DMA 1, Channel 0.  00 Reserved 01 Reserved 10 Reserved 11 DMA's Channel may be initiated by EPU
2–31 -	This field is reserved. Reserved

### 13.3.21 Topology Initiator Type n Register (DCFG\_CCSR\_TP\_ITYPn)

Each Initiator Type Topology Register provides one entry of a 64 entry lookup table.

Address: 1EE\_0000h base + 740h offset + (4d × i), where i=0d to 63d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	n	n	n	n	n	n	n	

**DCFG\_CCSR\_TP\_ITYPn field descriptions**

Field	Description
0–23 -	This field is reserved. Reserved
24–31 INIT_TYPE	Initiator Type. This field identifies the type of initiator (core or hardware accelerator) for this index. The lsb differentiates between an enabled and a disabled instance of the initiator. All encodings not listed below are reserved.  00h Simple initiator 02h Arm Cortex A53 (disabled) 03h Arm Cortex A53 (enabled)

### 13.3.22 Core Cluster n Topology Register (DCFG\_CCSR\_TP\_CLUSTERn)

Each Topology Cluster Register (TP\_CLUSTERn) contains four 6-bit fields, each of which is an index used for an initiator type lookup in a 64 entry initiator table implemented using the Topology Type Registers.

Address: 1EE\_0000h base + 844h offset + (8d × i), where i=0d to 0d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	EOC	IT_IDX_PC4								IT_IDX_PC3							
W									Reserved								
Reset	n	n	n	n	n	n	n	n	0	0	n	n	n	n	n	n	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	Reserved								IT_IDX_PC1								
W									Reserved								
Reset	0	0	n	n	n	n	n	n	0	0	n	n	n	n	n	n	

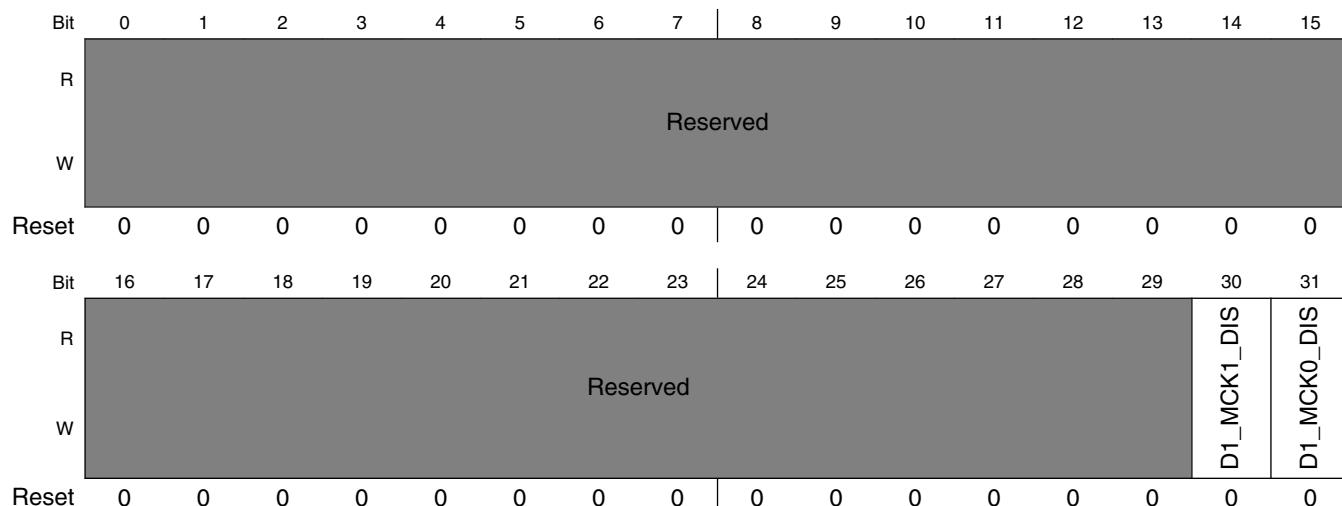
#### DCFG\_CCSR\_TP\_CLUSTERn field descriptions

Field	Description
0–1 EOC	End of Clusters - if the EOC field is non-zero, the register contains the information on the last cluster in the chip. 00 Not the last cluster 01,10,11 Last cluster in the chip
2–7 IT_IDX_PC4	Initiator Type Index for this cluster's fourth initiator Provides a 6-bit index for accessing an entry stored in one of the TP_ITYPn registers. This index selects which of the 64 TP_ITYPn registers contains the identification information for this initiator.
8–9 -	This field is reserved. Reserved
10–15 IT_IDX_PC3	Initiator Type Index for this cluster third initiator Provides a 6-bit index for accessing an entry stored in one of the TP_ITYPn registers. This index selects which of the 64 TP_ITYPn registers contains the identification information for this initiator.
16–17 -	This field is reserved. reserved
18–23 IT_IDX_PC2	Initiator Type Index for this cluster second initiator Provides a 6-bit index for accessing an entry stored in one of the TP_ITYPn registers. This index selects which of the 64 TP_ITYPn registers contains the identification information for this initiator.
24–25 -	This field is reserved. reserved
26–31 IT_IDX_PC1	Initiator Type Index for this cluster first initiator Provides a 6-bit index for accessing an entry stored in one of the TP_ITYPn registers. This index selects which of the 64 TP_ITYPn registers contains the identification information for this initiator.

### 13.3.23 DDR Clock Disable Register (DCFG\_CCSR\_DDRCLKDR)

DDRCLKDR allows for specific, unused clocks of the DDR Controller interface to be released to high impedance, thereby reducing power consumption.

Address: 1EE\_0000h base + E60h offset = 1EE\_0E60h



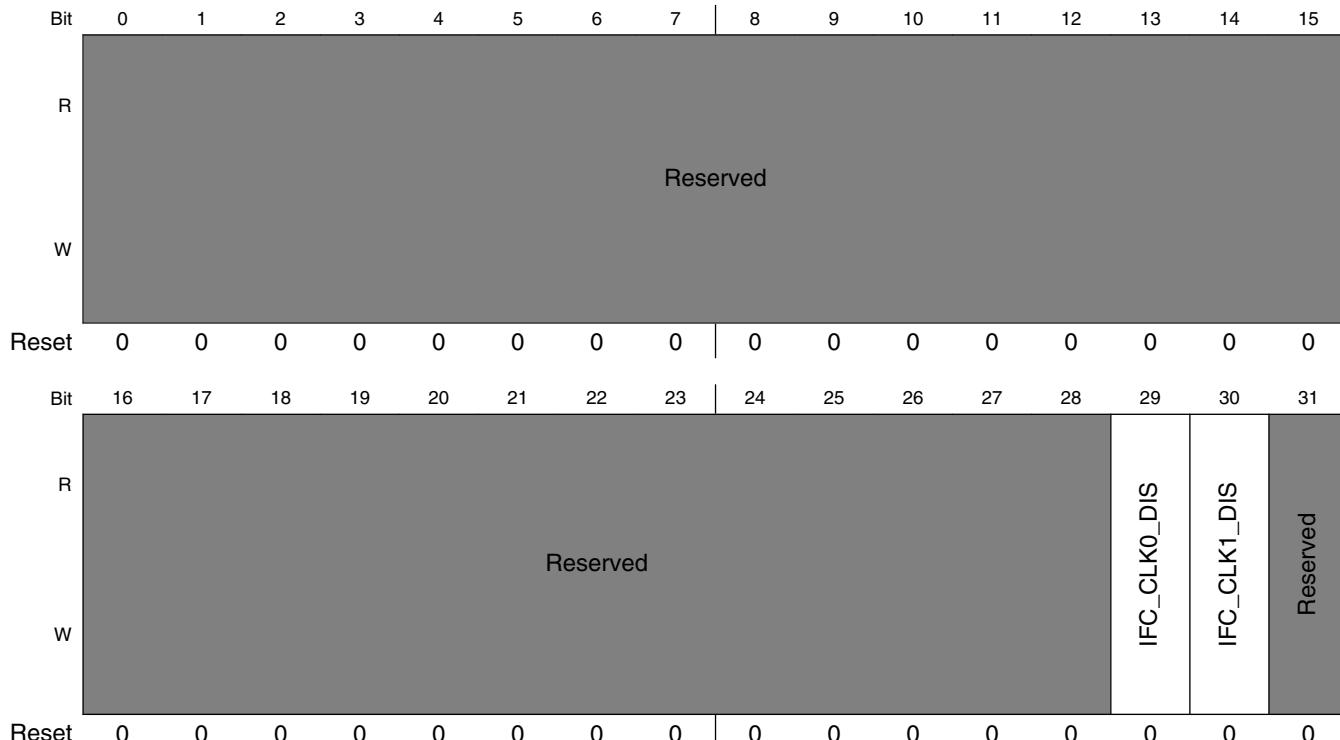
**DCFG\_CCSR\_DDRCLKDR field descriptions**

Field	Description
0–29 -	This field is reserved.
30 D1_MCK1_DIS	DDR Controller 1 clock 1 disable. The output is tri-stated, when disabled. 0b - D1_MCK[1] is enabled 1b - D1_MCK[1] is disabled
31 D1_MCK0_DIS	DDR Controller 1 clock 0 disable. The output is tri-stated, when disabled. 0b - D1_MCK[0] is enabled 1b - D1_MCK[0] is disabled

### 13.3.24 IFC Clock Disable Register (DCFG\_CCSR\_IFCCLKDR)

The IFCCLKDR allows for specific, unused clocks of the IFC interface to be not driven/high-impedance, thereby reducing power consumption.

Address: 1EE\_0000h base + E68h offset = 1EE\_0E68h



**DCFG\_CCSR\_IFCCLKDR field descriptions**

Field	Description
0–28 -	This field is reserved. Reserved
29 IFC_CLK0_DIS	IFC clock 0 disable. The output is not driven and in a high impedance state, when disabled. 0 IFC_CLK0 is enabled 1 IFC_CLK0 is disabled
30 IFC_CLK1_DIS	IFC clock 1 disable. The output is not driven and in a high impedance state, when disabled. 0 IFC_CLK1 is enabled 1 IFC_CLK1 is disabled
31 -	This field is reserved. Reserved

### 13.3.25 eSDHC Polarity Configuration Register (DCFG\_CCSR\_SDHPCR)

The SDHPCR allows for specific polarity control of the eSDHC input signals.

Address: 1EE\_0000h base + E80h offset = 1EE\_0E80h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CD_INV	WP_INV														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCFG\_CCSR\_SDHPCR field descriptions**

Field	Description
0 CD_INV	eSDHC Card Detect Invert 0 SDHC_CD signal to the eSDHC module has the same polarity as the I/O Pin. 1 SDHC_CD signal to the eSDHC module has inverted polarity in relation to the I/O Pin.
1 WP_INV	eSDHC Write Protect Invert 0 SDHC_WP signal to the eSDHC module has the same polarity as the I/O Pin. 1 SDHC_WP signal to the eSDHC module has inverted polarity in relation to the I/O Pin.
2-31 -	This field is reserved. Reserved



# **Chapter 14**

## **Run Control and Power Management (RCPM)**

### **14.1 Introduction**

This chapter provides the specification and programming model for the RCPM.

#### **14.1.1 Overview**

The Run Control and Power Management (RCPM) module communicates with embedded cores, coherency modules, and other device platform module to provide run control and power management functionality. The device can be placed into a range of low power states, via the RCPM (and its associated RCPM driver), to significantly reduce dynamic power consumption at the processor core, cluster, and device level. The RCPM also provides the functionality used to return processors, clusters, and device platform module to full operation in response to wake-up events such as external signals, timers, interrupts, and network traffic.

A complete whitepaper on QorIQ Power Management can be found at [QorIQ Power Management](#).

#### **14.1.2 The RCPM module as implemented on the chip**

This section provides details about how the RCPM module is integrated into this chip.

The RCPM module supports multithreaded cores; however, the A53 cores implemented on chip are single-threaded. The following table describes the mapping of the referenced RCPM threads and physical cores to the Arm Cortex-A53 cores:

**Table 14-1. RCPM/LS1043A thread and physical core *n* mapping**

RCPM thread n	LS1043A core	RCPM physical core n	LS1043A core
0	core 0	0	core 0
1	core 1	1	core 1
2	core 2	2	core 2
3	core 3	3	core 3

### 14.1.3 Power Management Features

- Software controlled power management, which minimizes power consumption of blocks when they are idle.
- Software-controlled power management states
  - Per core STANDBYWFI (PW15)
  - Per device LPM20 and SWLPM20
  - Per physical core-PH20
- Supports interrupt controller interrupt-based wake-up and the following wake-up sources:
  - Wake on internal timer event.
  - Wake on internal and external interrupt event.
- Enter software-controlled power state through core
- Core power management:
  - Independent power management control of each core
  - STANDBYWFI (PW15) state where:
    - Entry into PW15 mode core power management state via wait for interrupt instruction execution by the core
    - STANDBYWFI (PW15) state wake up source is captured in [Table 14-4](#)
- PH20 state where:
  - Entry into PH20 core power management state via RCPM PCPH20SETR for request to enter the core's PH20 state.
- Independent wake up from per thread:
  - Unmasked interrupt request
  - Unmasked critical interrupt
  - Debug interrupt
- Cluster power management:

- WFI instruction by all the cores would initiate L2 Cache of the cluster to go into STANDBY mode.
- The entry and exit mode of cluster power management is captured in [Modes Entry and Exit for Power Management](#)
- Device Power Management:
  - LPM20 state where:
    - All cores are in STANDBYWFI (PW15) state.
    - All cores are in PH20 state.
    - Platform clock is disabled.
    - Entry into LPM20 via setting POWMGTCSR[LPM20\_REQ] as well as WFI Instruction execution from all the cores.
    - LPM20 state wake up source is captured in [Table 14-5](#)
  - SWLPM20 state where:
    - All cores are in STANDBYWFI (PW15) state.
    - Platform clock is disabled.
    - Entry into SWLPM20 via WFI Instruction execution from all the cores.
    - SWLPM20 state wake up source are the same as is captured in [Table 14-5](#).
  - Independent Device wake up from:
    - Unmasked interrupt configured in GIC-400 and RCPM registers
    - Unmasked critical interrupt

#### 14.1.4 Power Management States

The RCPM module supports the following Power Management modes of operation:

- Power management states
  - Core(s)
    - Core(s) Full On state
    - Core(s) in STANDBYWFI (PW15) state
    - Core(s) in PH20Core(s) in PH20 state
  - Device
    - Device Full On state
    - Device in LPM20 state
    - Device in SWLPM20 state
- Reset modes

The different states available are summarized in [Table 14-2](#) through [Table 14-5](#).

### 14.1.4.1 Power Management State Summary

The following table summarizes the core power management states. For detailed description of power management states, refer [Modes Entry and Exit for Power Management](#).

**Table 14-2. Core and Cluster Power Management State Summary**

Power Management State Name	Resumable <sup>1</sup>	Services Snoops	Device Clocks	Core Clocks	State Retained	Description
Full On	NA	Y	Y	Y	Y	The core referred to is currently not in the STANDBYWFI (PW15) state.
STANDBYWFI (PW15)	Y	Y	Y	N	Y	The core referred to is currently in the STANDBYWFI (PW15) state.
PH20	Y	N	Y	N	Y	

1. A resumable power state in the core indicates that the core can exit from a power managed state and return to Full On without the core(s) being reset.

[Table 14-3](#) summarizes the device RCPM modes.

**Table 14-3. Device Power Management State Summary**

Power Management State Name	Resumable <sup>1</sup>	Services Snoops	Device Clocks	Core Clocks	Description
Full On	NA	Y	Y	Y	Platform clock to all modules is not gated.
LPM20	Y	N	N	N	Platform clock to all modules is gated off.
SWLPM20	Y	N	N	N (WFI)	Platform clock to all modules is gated off.

1. A resumable power state for the device indicates that the device can exit from a power managed state and return to Full On without the device being reset.

### 14.1.5 Modes Entry and Exit for Power Management

[Table 14-4](#) summarizes the entry and exit for the core power management modes.

**Table 14-4. Core and Cluster Power Management States: Entry and Exit**

Power Management State Name	Entry via ...	Exit via ...	
		Temporary exit	Permanent exit
STANDBYWFI (PW15)	<ul style="list-style-type: none"> <li>Core Executes WFI (wait for interrupt) instruction.</li> </ul>	<ul style="list-style-type: none"> <li>A snoop request that must be serviced by the core L1 data cache.</li> <li>A cache or TLB maintenance operation that must be serviced by the core L1 instruction cache, data cache, or TLB.</li> <li>A core debug halt request.</li> <li>An APB access to the debug or trace registers residing in the core power domain.</li> </ul>	<ul style="list-style-type: none"> <li>Core warm reset or any device-level reset</li> <li>Any interrupt request.<sup>1</sup></li> <li>A core debug halt request.<sup>2</sup></li> <li>Debug interrupt without masking</li> <li>Wake on IRQ[0-11]</li> <li>Wake on software interrupt</li> <li>Wake on peripheral interrupt</li> <li>Wake on HRESET</li> <li>Wake on PORESET</li> <li>Wake on debug halt</li> <li>Wake on debug interrupt</li> </ul>
PH20	<ul style="list-style-type: none"> <li>Privileged software set corresponding bit in RCPM PCPH20SETR to 1.</li> </ul> <p>Core PH20 (PGSR) operation can be triggered by the following mechanism:</p> <ol style="list-style-type: none"> <li>Set CPUECTLR[2:0] CPU retention control to a non-zero value. Refer Arm® Cortex®-A53 Technical Reference Manual for more information.</li> <li>Set CPUECTLR[6] SMPEN to 1. The cluster power management controller uses the state of this bit to decide whether to put the core into retention (equals to one) or powerdown&gt;equals to zero).</li> <li>Write to generic timer control register SYS_Counter_CNTCR to enable the Arm counter CNTVALUEB[63:0].</li> <li>Write to generic timer control register SYS_Counter_CNTCR to enable the Arm counter CNTVALUEB[63:0]</li> <li>Set the RETREQn bit of corresponding core in SCFG_RETREQCR register for RETENTION_REQ_EN at arm_cluster to be set.</li> <li>Set the PC_PH20_REQ bit in RCPM_PCPH20SETR register through Software.</li> <li>Execute WFI Instruction of the Core.</li> </ol>		<p>The following core wake up events will unconditionally wake up core from power management state via core_wakeup_req signal.</p> <ul style="list-style-type: none"> <li>GIC core warm reset request deassertion</li> <li>Core debug halt request</li> <li>Debug interrupt without masking</li> <li>Wake on IRQ[0-11]</li> <li>Wake on peripheral interrupt</li> <li>Wake on HRESET</li> <li>Wake on PORESET</li> <li>Wake on debug halt</li> <li>Wake on debug interrupt</li> <li>Wake on interrupt with clearing PCPH20REQ in register PCPH20CLRR</li> <li>Wake on FlexTimer interrupt, if programmed</li> <li>Wake on IIC interrupt, IIC slave address matching, if programmed</li> <li>Wake on UART interrupt, UART data reception, if programmed</li> <li>Wake on IRQ0 and IRQ2, if programmed</li> </ul>

**Table 14-4. Core and Cluster Power Management States: Entry and Exit**

Power Management State Name	Entry via ...	Exit via ...	
		Temporary exit	Permanent exit
	8. COP sends RETENTION_REQ and waits for RETENTION output from the core. 9. When RETENTION signal is asserted at COP boundary, Core state machine moves to PH20 state. 10. RCPM_PCPH20SR register corresponding bit should get set.		

1. Interrupts may still be masked in the GIC. Masking in the GIC prevents interrupt delivery to the core and therefore does not cause an exit.
2. For a core debug halt request to cause an exit from power management mode, the core must be operating at a frequency greater than the platform frequency.

### NOTE

The power management states of A53 core can be achieved with the software sequence only without the RCPM interaction.

The registers given below provides the different status of A53 core:

- TWAITSR0: Status shows core is in the WFI state
- POWMGTCsr: LPM20 request and status

**Table 14-5** summarizes the entry and exit for the device power management modes.

**Table 14-5. Device RCPM Mode: Entry and Exit for Power Management**

Device Power Management State Name	Entry via ...	Exit via ...
LPM20	LPM20 operation can be triggered by the following mechanism: <ul style="list-style-type: none"> <li>• Set CPUECTLR[2:0] CPU retention control to a non-zero value.</li> <li>• Set CPUECTLR[6] SMPEN to 1. The cluster power management controller uses the state of this bit to decide whether to put the core into retention (equals to one) or powerdown&gt;equals to zero).</li> <li>• Write to generic timer CNCTR register to enable the Arm counter CNTVALUEB[63:0].</li> <li>• Set the SCFG_COREPMCR[WFL2].</li> <li>• Set the SCFG_RETREQCR[RETREQn] of corresponding core for             </li> </ul>	<ul style="list-style-type: none"> <li>• Wake on all FMan MACs - Magic Packet, if programmed</li> <li>• Wake on IIC1, if programmed</li> <li>• Wake on LPUART1, if programmed</li> <li>• Wake on FlexTimer1, if programmed</li> <li>• Wake on GPIO, if programmed</li> <li>• Wake on FMan, if programmed</li> <li>• Wake on IRQ pins</li> <li>• PORESET</li> </ul>

**Table 14-5. Device RCPM Mode: Entry and Exit for Power Management**

Device Power Management State Name	Entry via ...	Exit via ...
	<p>RETENTION_REQ_EN at Arm_cluster to be set.</p> <ul style="list-style-type: none"> <li>• Set RCPM_POWMGTCSR[LPM20 REQ].</li> <li>• Execute WFI instruction on each core.</li> </ul> <p>After these programming ASLEEP pin should get asserted.</p> <p><b>NOTE:</b> When any core is in the WFI state, the device cannot be configured in the LPM20 state until the core comes out of the WFI state. To configure the device in sleep state (LPM20) when some cores are already in the WFI state, the software should first bring the cores out of WFI through software-generated interrupt. It should then wait for the interrupt to be cleared and the WFI status bits to be cleared in the SCFG_LPMCSR register before following the LPM20 sequence to put the device in sleep state.</p> <p><b>NOTE:</b> When cores(core) are in boot hold-off state, the device cannot be put in sleep state. To follow the LPM20 sequence in such cases, these cores should be first brought out from the boot hold-off states.</p>	

#### 14.1.5.1 SWLPM20 Entry Sequence – System Software

The system software is responsible for performing the following actions:

1. Software quiesces all external devices/internal modules through configuration. This ensures that no master or peripheral (for example I2C, UART, WDOG, FlexTimer, and eSDHC) is active while the device is attempting to enter LPM20. For system implementations running on Linux, this is typically covered by the Linux drivers of these peripherals. However, if standard system software is not used, this process to quiesce the device must be implemented.
2. The modules which are not to be powered down are listed in IPPDEXPCR. (This action is typically performed by standard Linux).

3. Redirect LPM20 wake-up conditions/interrupts to core 0. Note that for systems that are running Linux, the kernel may already perform this function of redirecting all interrupts to core 0. All of the secondary cores (other than the last active core, core 0) will no longer be available after the next step.
4. Put the secondary cores (other than the last active core, core 0) into WFI state through PSCI.
5. Wake from LPM20 due to PCI express events is only supported by a side band signal which is connected to IRQ or GPIO only. Other LPM20 exit conditions are listed in section [Table 14-5](#).
6. The last core calls the PSCI function CPU\_SUSPEND[system.power-down], which implements the LPM20 sequence.

#### **14.1.5.2 SWLPM20 Device Specific Entry and Exit Sequence**

The detailed SWLPM20 entry and exit sequence for the device is as follows:

**NOTE**

The code must be executed at EL3.

1. Mask interrupts at the core (DAIF = 0x3C0).
2. Disable D-Cache. Clear C bit in SCTRLR\_EL3 (Arm core register).
3. Invalidate/cln L1 D-Cache.
4. Set WFIL2\_EN in COREPMCR.
5. Write (32-bit) 0x80000000 to address 0x20170000.
6. Set IRQ bit in SCR\_EL3.
7. Read (32-bit) IPPDEXPCR0 @ 0x01EE2140
  - if IPPDEXPCR[3] = 1, exclusion\_mask\_1 = 0x00000080
  - if IPPDEXPCR[19] = 1, exclusion\_mask\_2 |= 0x00000002
  - if IPPDEXPCR[18] = 1, exclusion\_mask\_2 |= 0x00020000
  - if IPPDEXPCR[17] = 1, exclusion\_mask\_2 |= 0x00000400
  - if IPPDEXPCR[16] = 1, exclusion\_mask\_2 |= 0x02000000
  - if IPPDEXPCR[6] = 1, exclusion\_mask\_2 |= 0x00400000

exclusion\_mask\_2 indicate those modules which are already disabled.
8. Write (32-bit) 0xA000C201 to address 0x20170008.
9. If exclusion\_mask\_1 is 0, write (32-bit) the value 0x00000080 to address 0x2017000C.
10. Write (32-bit) the value 0x000C0000 to address 0x20170010.
11. Write (32-bit) the value 0x38000000 to address 0x20170014.
12. Write (32-bit) the value (0x10A33BFC &~exclusion\_mask\_2) to address 0x20170028.

13. Read (32-bit) address 0x20170018. Poll this address until the value 0xA000C201 is read.
14. If exclusion\_mask\_1 is 0, read/poll (32-bit) on address 0x2017001C until the value 0x00000080 is returned.
15. Read (32-bit) address 0x20170020. Poll this address until the value 0x000C0000 is returned.
16. Read (32-bit) address 0x20170024. Poll this address until the value 0x38000000 is returned.
17. Read (32-bit) address 0x2017002C. Poll this address until the value (0x10A33BFC &~exclusion\_mask\_2) is returned .  
exclusion\_mask\_2 indicate modules which are already disabled.
18. Save the value of register DEVDISR1 and write (32-bit) the new value of 0xA0C3C201 to address 0x01EE0070.
19. Save the value of register DEVDISR2 and write (32-bit) the new value of (0xCC0C0080 &~exclusion\_mask\_1) to address 0x01EE0074.  
exclusion\_mask\_2 indicates module which are already disabled.
20. Save the value of register DEVDISR3 and write (32-bit) the new value of 0xE00C0000 to address 0x01EE0078.
21. Save the value of register DEVDISR4 and write (32-bit) the new value of 0x38000000 to address 0x01EE007C.
22. Save the value of register DEVDISR5 and write (32-bit) the new value of (0x10A33BFC &~exclusion\_mask\_2) to address 0x01EE0080.
23. Disable data pre-fetch in Arm core register CPUACTLR\_EL1.
24. Set register DDR\_TIMING\_CFG\_4 bits [2:0] = 0x2.

### NOTE

The following sequence pre-fetches lines into the instruction cache for execution. This is necessary because DDR is going into self-refresh, and the clock to the DDR controller will be stopped. The code is shown below.

```
// On input, the following registers are loaded with the specified values:  
  
// w13 = original contents of DEVDISR1  
// w14 = original contents of DEVDISR2  
// w15 = original contents of DEVDISR3  
// w16 = original contents of DEVDISR4  
// w17 = original contents of DEVDISR5
```

The general algorithm to shut down the DDR controller clock is provided below, this is implemented in the AArch64 assembly language code:

- Load all instructions into the I-Cache so there are no fetches from DDR (D-Cache has already been disabled).
- Put DDR into self-refresh.
- Request that the DDR controller interface complete all the transactions (quiesce).
- Poll until DDR controller interface is quiescent.
- Stop the clock to the DDR controller.
- Put the core into WFI.
- On exiting WFI, unwind the above.
- If retry count is exceeded while polling for the DDR controller interface to quiesce; put DDR out of self refresh and exit with an error.

```

// 4Kb aligned - prevents taking a page fault in the middle
.align 12
    mov x0, xzr      // initialize x0=0 (cache prefetch control)
    b touch_line_0   // prefetches first cache line
.start_line_0:
    mov x0, #1        // turn off cache line prefetch
    mov x2, 0x80000000 // put ddr in self-refresh - start
    ldr w3, [0x01080114]
    rev w4, w3
    orr w4, w4, w2
    rev w3, w4
    str w3, [0x01080114] // put ddr in self refresh - end
    orr w3, w5, 0x80000000 // request ddr cntlr interface quiesce - start
    rev w4, w3
    str w4, [0x20170028] // request ddr cntlr interface quiesce - end
    mov w3, 0x80000000
    rev w3, w3
    mov x2, 0x10000 // retry count for ddr cntlr interface polling
.touch_line_0:
    cbz x0, touch_line_1 // prefetches second cache line when x0=0
.start_line_1:
    ldr w1, [0x2017002C] // poll on ddr cntlr quiesce ack
    tst w1, w3
    b.ne if // ddr cntlr is quiescent, proceed
    subs x2, x2, #1
    b.gt start_line_1 // loop back and try again
    rev w4, w5 // timeout error on reaching here
    str w4, [0x20170028] // deassert the ddr cntlr quiesce request
    mov x0, #-1 // load error code
    b 2f // finish cleanup
1:
    str w4, [0x01EE0080] // disable ddr cntrlr clk
    wfi // power-down core
    rev w4, w5
    str w4, [0x01EE0080] // re-enable ddr cntlr clock
    str w4, [0x20170028] // deassert the ddr cntlr quiesce request
    nop
.touch_line_1:
    cbz x0, touch_line_2 // prefetches third cache line when x0=0
.start_line_2:
    ldr w1, [0x2017002C]
    tst w1, w3
    b.eq 2f
    nop
    b start_line_2
2:
    mov x2, 0x80000000

```

```

ldr w3, [0x01080114]
rev w4, w3
bic w4, w4, w2
rev w3, w4
mov x1, 0x10000           // ddr clock restart delay count
3:
subs x1, x1, #1
b.gt 3b      // delay for ddr clock to restart
str w3, [0x01080114]      // take ddr out-of self refresh
nop
touch_line_2:
cbz x0, touch_line_3    // prefetches fourth cache line when x0=0
start_line_3:
str w17, [0x01EE0080]    // reload DEVDISR5
str w16, [0x01EE007C]    // reload DEVDISR4
str w15, [0x01EE0078]    // reload DEVDISR3
str w14, [0x01EE0074]    // reload DEVDISR2
str w13, [0x01EE0070]    // reload DEVDISR1
str wzr, [0x20170028]   // deassert ip block quiesce request
str wzr, [0x20170014]   // deassert ip block quiesce request
str wzr, [0x20170010]   // deassert ip block quiesce request
str wzr, [0x2017000C]   // deassert ip block quiesce request
str wzr, [0x20170008]   // deassert ip block quiesce request
nop
nop
nop
nop
b continue_restart      // ddr is now functional, can leave cache-only region
touch_line_3:
cbz x0, start_line_0    // begin executing SWLPM20 sequence from cache
// execute here after ddr is back up

continue_restart:

```

25. write (32-bit) 0x0 to address 0x20170000.
26. write (32-bit) 0x0 to address 0x0157042C (clear WFIL2\_EN in COREPMCR).

## 14.1.6 Power Management States

This section provides functional explanation of the power management states summarized in the previous sections.

The privileged software can place the device in one of the following power states:

- STANDBYWFI (PW15) state
- PH20 (core retention)

**Table 14-6. Power state entry by register**

State	Register
core PH20	PCPH20SETR

- LPM20 (sleep)
- SWLPM20 (sleep)

### 14.1.6.1 STANDBYWFI (PW15) State

In STANDBYWFI (PW15) state, the core internally gates clocks within the core, significantly reducing the power consumption of the core. Snooping of the L1 and L2 are still supported and thus the data in the data cache is kept coherent. Interrupts directed to the Core are monitored by the GIC and cause core to exit from STANDBYWFI (PW15) state to allow the core to recognize and process the interrupt. STANDBYWFI (PW15) state is entered through execution of wait for interrupt instruction from the core.

The watchdog timer facilities are still enabled during STANDBYWFI (PW15) state.

### 14.1.6.2 PH20 State

Core PH20 state is core power gating with state retention.

### 14.1.6.3 LPM20 State

LPM20 is a device low power state. Prior entering LPM20, software is expected to quiesce all external devices/internal IPs through configuration. RCPM hardware is designed to implement sanity check for IPs idle status.

In LPM20 state, all cores and clusters are placed in their lowest power managed state - PH20. All other device modules in chip are clock gated, except I2C, GPIO, IRQ, UART, and FlexTimer so that only those modules which are required to wake up the device will still have a running clock.

The following figure describes the sleep flow for the chip.

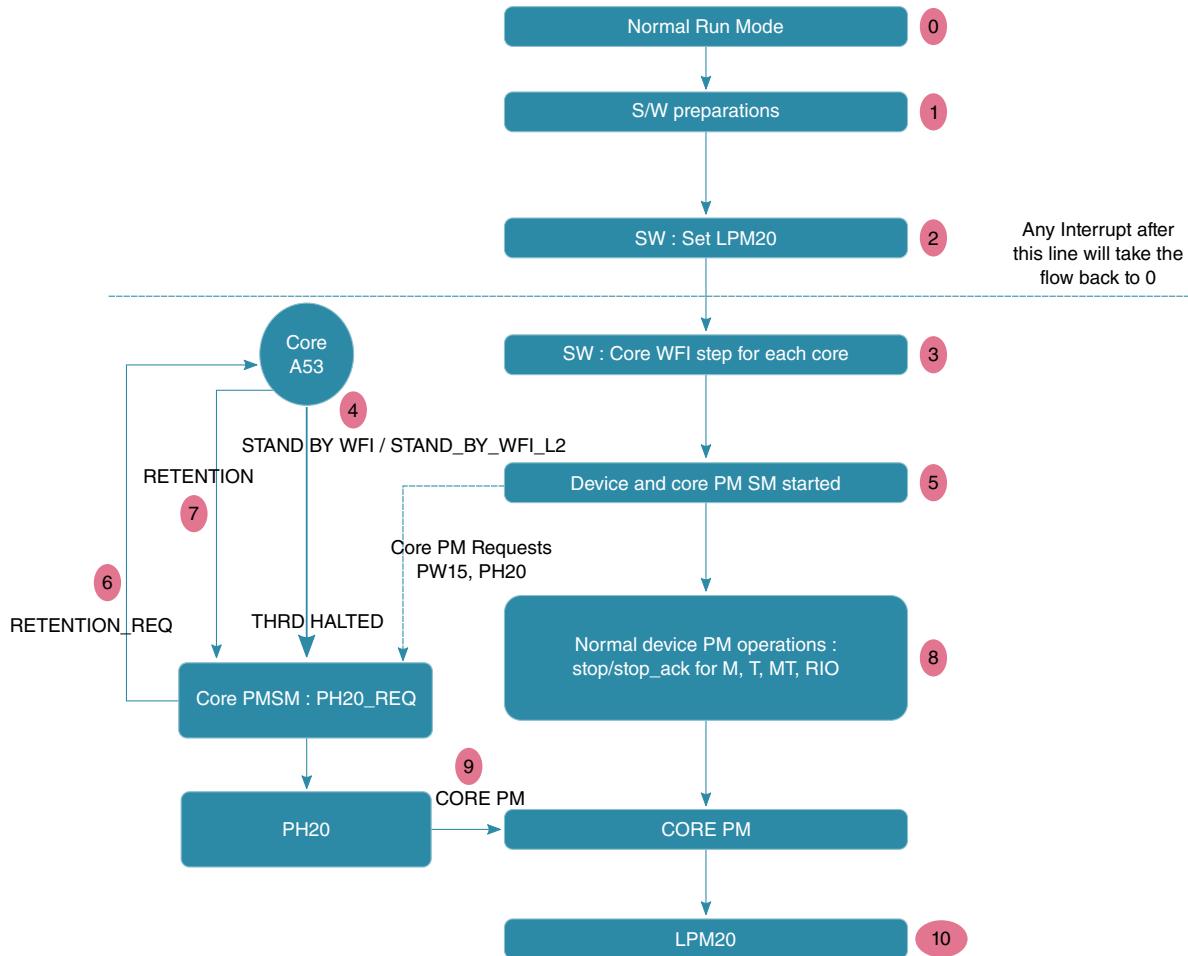


Figure 14-1. Device LPM20 operation sequence

#### 14.1.6.4 SWLPM20 State

SWLPM20 is a device low power state. Prior entering SWLPM20, software is expected to quiesce all external devices/internal IPs through configuration. RCPM hardware is designed to implement sanity check for IPs idle status.

In SWLPM20 state, all cores and clusters are placed in their lowest power managed state - PW15. All other device modules in chip are clock gated, except GPIO, IRQ, LPUART1, FlexTimer, DDR, FMAN\_MAC3, and, FMAN\_MAC4 so that only those modules which are required to wake up the device will still have a running clock.

## 14.1.7 Reset modes

This section describes the relationship between power management states and reset modes.

### 14.1.7.1 Power-On Reset and Hard Reset States

During power-on reset and hard reset, the following hardware structures are reset:

- All CCSR registers
- SoC power management state machine
- Core power management state machine
- Cluster power management state machine

## 14.2 External Signal Description

The table below provides the signal description of power management signals:

**Table 14-7. RCPM Detailed Signal Descriptions**

Signal	I/ O	Description
ASLEEP	O	Asleep. After negation of PORESET, ASLEEP is asserted until the device completes its power-on reset sequence and reaches its ready state.
		<b>State Meaning</b> Asserted- Indicates that the device is either still in its power-on reset sequence or it has reached a LPM20 (sleep) state after a power-down command is issued by software. Negated- The device is not in LPM20 (sleep) state. (It has either awoken from a power-down state, or has completed the POR sequence.)
		<b>Timing</b> Assertion- May occur at any time; may be asserted asynchronously to the input clocks. Negation- Negates synchronously with SYSCLK when leaving power-on sequence; otherwise negation is asynchronous.
RTC	I	Real Time Clock. Refer to the hardware specification of device for timing specification. The signal can be optionally used to clock the global timers in the programmable interrupt controller.
		<b>Timing</b> Assertion/Negation - See the hardware specification of device for specific timing information for this signal.

## 14.3 RCPM Memory Map/Register Definition

This section identifies power management resources that are not included as part of a processor core or platform IP.

**RCPM memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1EE_204C	Thread Wait status Register (RCPM_TWAITSR)	32	R	0000_0000h	<a href="#">14.3.1/535</a>
1EE_20D0	Physical Core PH20 Status Register (RCPM_PCPH20SR)	32	R	0000_0000h	<a href="#">14.3.2/536</a>
1EE_20D4	Physical Core PH20 Set Control Register (RCPM_PCPH20SETR)	32	R/W	0000_0000h	<a href="#">14.3.3/536</a>
1EE_20D8	Physical Core PH20 Clear Control Register (RCPM_PCPH20CLRR)	32	w1c	0000_0000h	<a href="#">14.3.4/537</a>
1EE_20DC	Physical Core PH20 Previous Status Register (RCPM_PCPH20PSR)	32	R	0000_0000h	<a href="#">14.3.5/539</a>
1EE_2130	Power Management Control and Status Register (RCPM_POWMGTCsr)	32	R/W	0000_0000h	<a href="#">14.3.6/539</a>
1EE_2140	IP Powerdown Exception Control Register (RCPM_IPPDEXPCR)	32	R/W	0000_0000h	<a href="#">14.3.7/541</a>
1EE_215C	nIRQOUT interrupt mask register (RCPM_nIRQOUTR)	32	R/W	0000_0000h	<a href="#">14.3.8/543</a>
1EE_216C	nFIQOUT Interrupt Register (RCPM_nFIQOUTR)	32	R/W	0000_0000h	<a href="#">14.3.9/543</a>

**14.3.1 Thread Wait status Register (RCPM\_TWAITSR)**

This register is used for reporting wait status per core.

Address: 1EE\_2000h base + 4Ch offset = 1EE\_204Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	T31_T0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**RCPM\_TWAITSR field descriptions**

Field	Description
0–31 T31_T0	Core STANDBYWFI (PW15) Status.  <b>NOTE:</b> Bit 31 corresponds to core 0, bit 30 corresponds to core 1, and so on.  0 Core is not in the STANDBYWFI (PW15) state 1 Core is in the STANDBYWFI (PW15) state

### 14.3.2 Physical Core PH20 Status Register (RCPM\_PCPH20SR)

This register is used for reporting PH20 status per physical core.

Address: 1EE\_2000h base + D0h offset = 1EE\_20D0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	PCn															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### RCPM\_PCPH20SR field descriptions

Field	Description				
0–31 PCn	<p>Physical core PH20 status.</p> <p><b>NOTE:</b> Bit 31 corresponds to Core 0, bit 30 corresponds to Core 1, and so on.</p> <table> <tr> <td>0</td> <td>Physical core is not in its PH20 state</td> </tr> <tr> <td>1</td> <td>Physical core is in its PH20 state</td> </tr> </table>	0	Physical core is not in its PH20 state	1	Physical core is in its PH20 state
0	Physical core is not in its PH20 state				
1	Physical core is in its PH20 state				

### 14.3.3 Physical Core PH20 Set Control Register (RCPM\_PCPH20SETR)

This register is used for requesting that a physical core be placed in the PH20 state. The true value read from the register means there is a pending physical core PH20 request.

#### NOTE

The RCPM module as implemented on the chip describes the cores that are implemented on the chip. In the following register description, any bits associated with unimplemented cores are reserved.

Address: 1EE\_2000h base + D4h offset = 1EE\_20D4h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	PC_PH20_REQ															
W																																

#### RCPM\_PCPH20SETR field descriptions

Field	Description
0–31 PC_PH20_REQ	Write one to trigger physical core PH20 request. This bit is clear by interrupt event or write-one event on PCPH20CLRRn register.

**RCPM\_PCPH20SETR field descriptions (continued)**

Field	Description
xxxxxxxx_xxxxxxxx_xxxxxxxx_xxxxxxx1	Request to put physical core 0 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_xxxxxx1x	Request to put physical core 1 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_xxxxxx1xx	Request to put physical core 2 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_xxxx1xxx	Request to put physical core 3 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_xxx1xxxx	Request to put physical core 4 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_xx1xxxxxx	Request to put physical core 5 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_x1xxxxxx	Request to put physical core 6 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxxx_1xxxxxxx	Request to put physical core 7 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxxx1_xxxxxxxx	Request to put physical core 8 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxx1x_xxxxxxxx	Request to put physical core 9 in PH20 state.
xxxxxxxx_xxxxxxxx_xxxxxx1xx_xxxxxxxx	Request to put physical core 10 in PH20 state.
xxxxxxxx_xxxxxxxx_xxx1xxx_xxxxxxxx	Request to put physical core 11 in PH20 state.
xxxxxxxx_xxxxxxxx_xxx1xxxx_xxxxxxxx	Request to put physical core 12 in PH20 state.
xxxxxxxx_xxxxxxxx_xx1xxxxx_xxxxxxxx	Request to put physical core 13 in PH20 state.
xxxxxxxx_xxxxxxxx_x1xxxxxx_xxxxxxxx	Request to put physical core 14 in PH20 state.
xxxxxxxx_xxxxxxx1_xxxxxxxx_xxxxxxxx	Request to put physical core 15 in PH20 state.
xxxxxxxx_xxxxxx1x_xxxxxxxx_xxxxxxxx	Request to put physical core 16 in PH20 state.
xxxxxxxx_xxxxxx1xx_xxxxxxxx_xxxxxxxx	Request to put physical core 17 in PH20 state.
xxxxxxxx_xxxxxx1xx_xxxxxxxx_xxxxxxxx	Request to put physical core 18 in PH20 state.
xxxxxxxx_xxx1xxx_xxxxxxxx_xxxxxxxx	Request to put physical core 19 in PH20 state.
xxxxxxxx_xxx1xxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 20 in PH20 state.
xxxxxxxx_xx1xxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 21 in PH20 state.
xxxxxxxx_x1xxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 22 in PH20 state.
xxxxxxxx_1xxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 23 in PH20 state.
xxxxxxxx1_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 24 in PH20 state.
xxxxx1x_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 25 in PH20 state.
xxxxx1xx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 26 in PH20 state.
xxxx1xxx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 27 in PH20 state.
xx1xxxx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 28 in PH20 state.
x1xxxxx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 29 in PH20 state.
1xxxxxxx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 30 in PH20 state.
1xxxxxxx_xxxxxxxx_xxxxxxxx_xxxxxxxx	Request to put physical core 31 in PH20 state.

#### 14.3.4 Physical Core PH20 Clear Control Register (RCPM\_PCPH20CLRR)

This register is used for waking up the core placed in the PH20 state. The true value read from the register only means there is a pending physical core PH20 request.

##### NOTE

The RCPM module as implemented on the chip describes the cores that are implemented on the chip. In the following register description, any bits associated with unimplemented cores are reserved.

## RCPM Memory Map/Register Definition

Address: 1EE\_2000h base + D8h offset = 1EE\_20D8h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## RCPM\_PCPH20CLRR field descriptions

Field	Description
0–31 CP_PH20_REQ	<p>Write one to cancel physical core PH20 request. The bit is set by write-one-event to PCPH20SETR register. This bit is also clear by interrupt event. The read true value of the register bit means there is pending Physical Core PH20 request for that core.</p> <p>xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxx1  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxx1x  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxx1xx  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxx1xxx  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxx1xxxx  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xx1xxxxxx  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_x1xxxxxx  xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_1xxxxxxx  xxxxxxxx_xxxxxxxxxx_xxxxxxx1_xxxxxxxxxx  xxxxxxxx_xxxxxxxxxx_xxxxxx1xx_xxxxxxxxxx  xxxxxxxx_xxxxxxxxxx_xxxx1xxx_xxxxxxxxxx  xxxxxxxx_xxxxxxxxxx_xxx1xxxx_xxxxxxxxxx  xxxxxxxx_xxxxxxxxxx_xx1xxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxxxxxx_x1xxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxxxxxx_1xxxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxxxxxx1_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxxx1x_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxxx1xx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  xxxx1xxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  xxx1xxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  xx1xxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  x1xxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx  1xxxxxxxx_xxxxxxxxxx_xxxxxxxxxx_xxxxxxxxxx</p> <p>core0 is receiving a PH20 request from PCPH20SETR  core1 is receiving a PH20 request from PCPH20SETR  core2 is receiving a PH20 request from PCPH20SETR  core3 is receiving a PH20 request from PCPH20SETR  core4 is receiving a PH20 request from PCPH20SETR  core5 is receiving a PH20 request from PCPH20SETR  core6 is receiving a PH20 request from PCPH20SETR  core7 is receiving a PH20 request from PCPH20SETR  core8 is receiving a PH20 request from PCPH20SETR  core9 is receiving a PH20 request from PCPH20SETR  core10 is receiving a PH20 request from PCPH20SETR  core11 is receiving a PH20 request from PCPH20SETR  core12 is receiving a PH20 request from PCPH20SETR  core13 is receiving a PH20 request from PCPH20SETR  core14 is receiving a PH20 request from PCPH20SETR  core15 is receiving a PH20 request from PCPH20SETR  core16 is receiving a PH20 request from PCPH20SETR  core17 is receiving a PH20 request from PCPH20SETR  core18 is receiving a PH20 request from PCPH20SETR  core19 is receiving a PH20 request from PCPH20SETR  core20 is receiving a PH20 request from PCPH20SETR  core21 is receiving a PH20 request from PCPH20SETR  core22 is receiving a PH20 request from PCPH20SETR  core23 is receiving a PH20 request from PCPH20SETR  core24 is receiving a PH20 request from PCPH20SETR  core25 is receiving a PH20 request from PCPH20SETR  core26 is receiving a PH20 request from PCPH20SETR  core27 is receiving a PH20 request from PCPH20SETR  core28 is receiving a PH20 request from PCPH20SETR  core29 is receiving a PH20 request from PCPH20SETR  core30 is receiving a PH20 request from PCPH20SETR  core31 is receiving a PH20 request from PCPH20SETR</p>

### 14.3.5 Physical Core PH20 Previous Status Register (RCPM\_PCPH20PSR)

This register is used for reporting previous PH20 status per physical core. It is used by software to know which power saving state it was in before wake up by interrupt.

Address: 1EE\_2000h base + DCh offset = 1EE\_20DCh

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	PC_P_PH20_n																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### RCPM\_PCPH20PSR field descriptions

Field	Description
0–31 PC_P_PH20_n	Physical core PH20 previous status. The bit is set when the corresponding PCPH20SR bit transitions from 1 to 0 due to an interrupt.  <b>NOTE:</b> Bit 31 corresponds to Core 0, bit 30 corresponds to Core 1, and so on.  0 Physical core was not in the PH20 state before wake up by interrupt. 1 Physical core was in the PH20 state before wake up by interrupt.

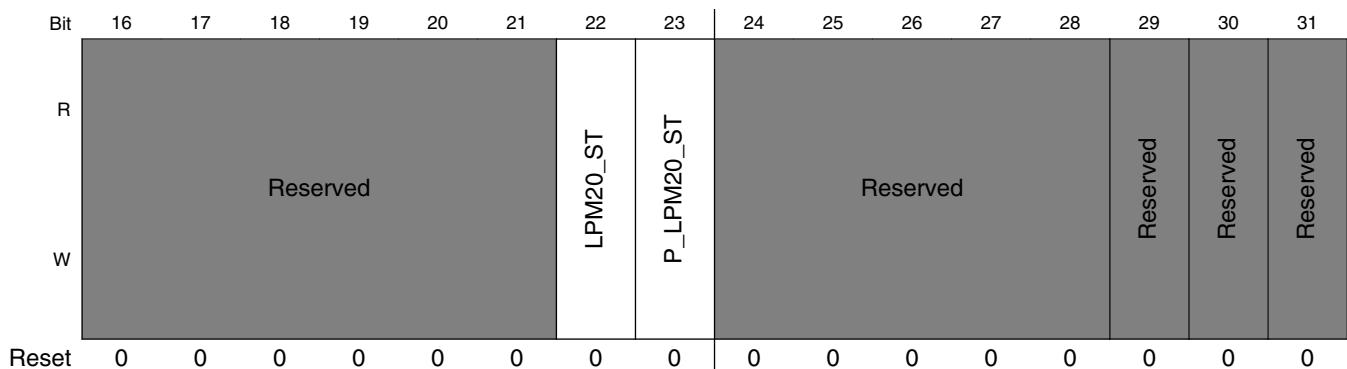
### 14.3.6 Power Management Control and Status Register (RCPM\_POWMGTCsr)

This register is used for entering the chip's LPM state. In addition, it provides status indicating successful entry into the corresponding power management state.

The following table describes this register's bit settings.

Address: 1EE\_2000h base + 130h offset = 1EE\_2130h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SD_PD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Reserved															
	LPM20_REQ															
	Reserved															
	Reserved															

**RCPM\_POWMGTCsr field descriptions**

Field	Description
0 SD_PD	SerDes and Protocol Converter Powerdown Control 0 SerDes and Protocol Converter are not powerdown in LPM state. 1 SerDes and Protocol Converter are powerdown in LPM state for further power saving. If this bit is set, after system wake up from LPM state, SerDes has to go through the training sequence to return back to function.
1–10 -	This field is reserved. Reserved
11 LPM20_REQ	LPM20 State Request. This bit is clear by interrupt event. 0 No request to put chip in LPM20 state 1 Request to place chip in LPM20 state.
12–13 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
15–21 -	This field is reserved. Reserved
22 LPM20_ST	LPM20 Status 0 Device is not attempting to reach LPM20 state 1 The chip is attempting to enter LPM20 state because POWMGTCsr[LPM20_REQ] is set.
23 P_LPM20_ST	Previous LPM20 Status. It is used by software to know which state the chip was in before being wake up. Before software set POWMGTCsr[LPM20_REQ], software w1c [P_LPM20_ST] bit is expected. 0 Device was not in LPM20 state before being wake up by interrupt. 1 Device was in LPM20 state before being wake up by interrupt. The bit is set when LPM20_ST transitions from 1 to 0 due to an interrupt. The bit is w1c.
24–28 -	This field is reserved. Reserved
29 -	This field is reserved. Reserved
30 -	This field is reserved. Reserved

*Table continues on the next page...*

**RCPM\_POWMGCSR field descriptions (continued)**

Field	Description
31 - Reserved	This field is reserved.

### 14.3.7 IP Powerdown Exception Control Register (RCPM\_IPPDEXPCR)

IPPDEXPCR provides a mechanism for excluding certain IP blocks from device LPM20 mode to make these IP blocks available as sources for wake-up events. For example,

- Wake on LAN (magic packet) - Requires excluding the Ethernet controller from device LPM20 mode
- Wake on GPIO - Requires excluding the GPIO controller from device LPM20 mode

Address: 1EE\_2000h base + 140h offset = 1EE\_2140h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MAC1_1	MAC1_2	MAC1_3	MAC1_4	MAC1_5	MAC1_6	Reserved	MAC1_9	Reserved	I2C1	LPUART1	FlexTimer1	OCRAM1			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								GPIO1	Reserved	FM1	Reserved				
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RCPM\_IPPDEXPCR field descriptions**

Field	Description
0 MAC1_1	Frame Manager 1 MAC1 powerdown exception 0 Frame Manager 1 MAC1 powerdown during device LPM20 1 Frame Manager 1 MAC1 is not powerdown during device LPM20
1 MAC1_2	Frame Manager 1 MAC2 powerdown exception 0 Frame Manager 1 MAC2 powerdown during device LPM20 1 Frame Manager 1 MAC2 is not powerdown during device LPM20
2 MAC1_3	Frame Manager 1 MAC3 powerdown exception

*Table continues on the next page...*

**RCPM\_IPPDEXPCR field descriptions (continued)**

Field	Description
	0 Frame Manager 1 MAC3 powerdown during device LPM20 1 Frame Manager 1 MAC3 is not powerdown during device LPM20
3 MAC1_4	Frame Manager 1 MAC4 powerdown exception 0 Frame Manager 1 MAC4 powerdown during device LPM20 1 Frame Manager 1 MAC4 is not powerdown during device LPM20
4 MAC1_5	Frame Manager 1 MAC5 powerdown exception 0 Frame Manager 1 MAC5 powerdown during device LPM20 1 Frame Manager 1 MAC5 is not powerdown during device LPM20
5 MAC1_6	Frame Manager 1 MAC6 powerdown exception 0 Frame Manager 1 MAC6 powerdown during device LPM20 1 Frame Manager 1 MAC6 is not powerdown during device LPM20
6–7 -	This field is reserved. Reserved
8 MAC1_9	Frame Manager 1 MAC9 powerdown exception 0 Frame Manager 1 MAC9 powerdown during device LPM20 1 Frame Manager 1 MAC9 is not powerdown during device LPM20
9–11 -	This field is reserved. Reserved
12 I2C1	I2C1 powerdown exception 0 I2C1 powerdown during device LPM20 1 I2C1 is not powerdown during device LPM20
13 LPUART1	LPUART1 powerdown exception 0 LPUART1 powerdown during device LPM20 1 LPUART1 is not powerdown during device LPM20
14 FlexTimer1	FlexTimer1 powerdown exception 0 FlexTimer1 powerdown during device LPM20 1 FlexTimer1 is not powerdown during device LPM20
15 OCRAM1	OCRAM1 powerdown exception 0 OCRAM 1 powerdown during device LPM20 1 OCRAM 1 is not powerdown during device LPM20
16–24 -	This field is reserved. Reserved
25 GPIO1	GPIO powerdown exception 0 GPIO powerdown during device LPM20 1 GPIO is not powerdown during device LPM20
26–27 -	This field is reserved. Reserved
28 FM1	Frame Manager 1 powerdown exception.

*Table continues on the next page...*

## **RCPM\_IPPDEXPCR field descriptions (continued)**

Field	Description
	0 FM1 powerdown during device LPM20 1 FM1 is not powerdown during device LPM20
29–31	This field is reserved.
-	Reserved

#### 14.3.8 nIRQOUT interrupt mask register (RCPM\_nIRQOUTR)

The register masks the nIROOUT Interrupt from GIC-400 for Sleep/LPM20 mode.

Address: 1EE 2000h base + 15Ch offset = 1EE 215Ch

## RCPM nIRQOUTR field descriptions

Field	Description
0-31 IMO	Interrupt mask core 0-31. <b>NOTE:</b> Bit 31 corresponds to Core 0, bit 30 corresponds to Core 1, and so on.

### 14.3.9 nFIQOUT Interrupt Register (RCPM\_nFIQOUTR)

The register masks the nFIQOUT interrupt from GIC-400 for Sleep/LPM20 mode. The LPM20 FSM (and software sequence in case of SWLPM20) recognizes the corresponding core interrupt from GIC-400 and initiates a wake-up sequence provided the interrupt mask is not set.

Address: 1EE 2000h base + 16Ch offset = 1EE 216Ch

## RCPM nFIQOUTR field descriptions

Field	Description
0-31 IMO	Interrupt mask core 0-31.

**RCPM\_nFIQOUTR field descriptions (continued)**

Field	Description
<b>NOTE:</b> Bit 31 corresponds to Core 0, bit 30 corresponds to Core 1, and so on.	

## 14.4 RCPM functional description

The RCPM supports minimizing the power consumption through software controlled power management states - PH20 for individual cores (STANDBYWFI (PW15)) and the device (LPM20).

RCPM can handshake with various IP at the SoC level to gracefully stop the IP traffic and direct the memory controller to put DDR into self-refresh mode (if enabled).

The RCPM allows several wake-up event sources to exit low power state (internal timer, internal and external interrupts).

The wake-up events are mapped to interrupt controller interrupts to generate a wake-up interrupt to the core. For information on operation entry or exit, refer [Modes Entry and Exit for Power Management](#).

# Chapter 15

## QUICC Engine Block

### 15.1 Introduction

The *QUICC Engine<sup>TM</sup> Block Reference Manual with Protocol Interworking* (QEIWRM) describes all the functional units of the QUICC Engine block and must be used in conjunction with this device manual and this chapter.

The QEIWRM is a superset manual which includes some information not relevant to the device. This chapter serves as both a general overview of the QUICC Engine block and a guide to the specific implementation of the QUICC Engine block on the device.

- [QUICC Engine block](#), gives a general overview of the QUICC Engine architecture and communication peripherals.
- [QUICC Engine implementation details](#), lists the chapters that do apply. Implementation-specific details for some chapters follow.

### 15.2 QUICC Engine block

#### NOTE

Based on the microcode RAM package that is used, some of the features listed below may not be applicable. See the table below for the different features offered in each microcode configuration.

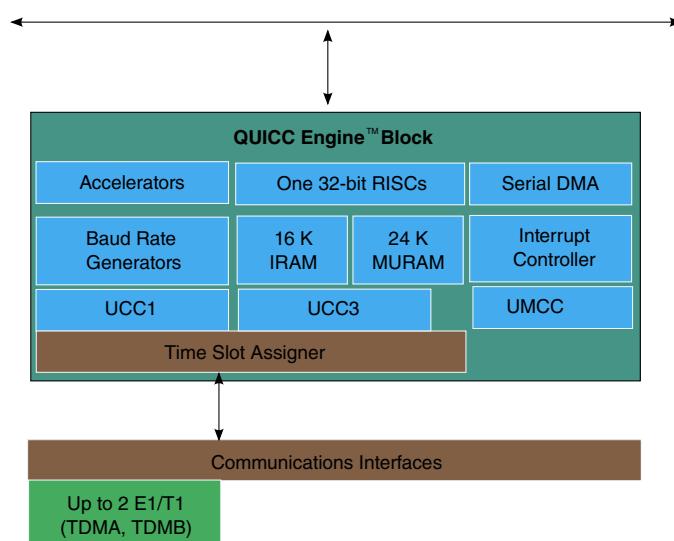
The QUICC Engine is a versatile communications complex that integrates several communications peripheral controllers. It provides on-chip system design for a variety of applications, particularly in communications and networking systems. The QUICC Engine has the following features:

- 32-bit RISC controller for flexible support of the communications peripherals
- Serial DMA channel for receive and transmit on all serial channels

## QUICC Engine implementation details

- Universal communication controllers (UCCs) supporting the following protocols and interfaces (not all of them simultaneously):
  - HDLC and Transparent controllers up to 50 Mbit/s full-duplex; HDLC bus up to 10 Mbit/s
  - UART
  - BISYNC up to 2 Mbit/s
  - UMCC
- TDM interfaces, with T1/E1/J1 serial interfaces
- 256 channels of HDLC/Transparent per UCC
- RAM-based microcode

The figure below shows the internal architecture and the interfaces provided by the QUICC Engine block on the device. A RAM is used to store parameters for the RISC engine. The instruction RAM is used to optionally run additional code.



**Figure 15-1. QUICC Engine block architectural block diagram**

## 15.3 QUICC Engine implementation details

The table below lists the chapters from the *QUICC Engine™ Block Reference Manual with Protocol Interworking* (QEIWRM).

Not all chapters of the QEIWRM apply to this device and those that do apply may have application differences. Use this overview section as a guide for these application differences.

While using the QEIWRM, note this implementation-specific information in general:

- Two UCCs available-UCC1 and UCC3
- Up to two TDM ports
- UMCC

The table below lists the chapters from the *QUICC Engine Block Reference Manual with Protocol Interworking* (QEWRM) and the chapters with implementation differences.

**Table 15-1. QEWRM chapters and implementation**

Chapter Name	Implementation
Part I, "Introduction"	
System Interface	Applies to this device Refer <a href="#">System interface</a> for implementation
Configuration	Applies to this device Refer <a href="#">Configuration</a> for implementation
Multiplexing and Timers	Applies to this device Refer <a href="#">Multiplexing and timers</a> for implementation
Part II, "Unified Communication Controllers (UCCs)"	
Unified Communications Controllers (UCCs)	Applies to this device Refer <a href="#">Unified communications controllers (UCCs)</a> for implementation
UCC for Fast Protocols	Applies to this device Refer <a href="#">UCC for fast protocols</a> for implementation
UCC Ethernet Controller (UEC)	Does not apply to this device
IEEE Standard 1588 Assist	Does not apply to this device
UTOPIA POS Bus Controller (UPC)	Does not apply to this device
ATM Adaptation Layer 2	Does not apply to this device
UCC POS Controller (UPOS)	Does not apply to this device
ATM Controller AAL0, AAL1, and AAL5	Does not apply to this device
HDLC Controller	Applies to this device Refer <a href="#">HDLC controller</a> for implementation
Transparent Controller	Applies to this device Refer <a href="#">Transparent controller</a> for implementation
UCC for Slow Protocols	Applies to this device Refer <a href="#">UCC for slow protocols</a> for implementation
UART Mode	Applies to this device
UCC UART PROFIBUS MODE	Applies to this device.
Serial Peripheral Interface (SPI)	Does not apply to this device
Universal Serial Bus Controller	Does not apply to this device
BISYNC Mode	Applies to this device Refer <a href="#">BISYNC mode</a> for implementation
Part III, "Time Division Multiplex Support (TDM)" <sup>1</sup>	
Serial Interface with Time-Slot Assigner	Applies to this device

*Table continues on the next page...*

**Table 15-1. QEIWRM chapters and implementation (continued)**

Chapter Name	Implementation
	Refer <a href="#">Serial interface with time-slot assigner</a> for implementation
Multi-Channel Controller (MCC)	Does not apply to this device
Multi-Channel Controller on UCC (UMCC)	Applies to this device Refer <a href="#">Multi-channel controller on UCC (UMCC)</a>
QUICC Multi-Channel Controller (QMC)	Does not apply to this device
Point-to-Point Protocol (PPP)	Does not apply to this device
Serial ATM Microcode	Does not apply to this device
Inverse Multiplexing for ATM (IMA)	Does not apply to this device
ATM AAL1 Circuit Emulation Service	Does not apply to this device
Part IV, "Multiprotocol Interworking"	
Frame Parse and Lookup	Does not apply to this device
Interworking Introduction	Does not apply to this device
Protocol Interworking Programming Model	Does not apply to this device
Virtual Port	Does not apply to this device
IP Reassembly Full	Does not apply to this device
IPv4/UDP Header Compression	Does not apply to this device
IPsec Microcode Package	Does not apply to this device
Part V, "Switching Functionality"	
Enhanced MSP Microcode	Does not apply to this device
L2 Ethernet Switch	Does not apply to this device

1. The TDM can only work with an external sync and external clock; other options are not supported.

The following subsections include device-specific details for the given chapters of the QEIWRM.

### 15.3.1 System interface

Of the UCCs, only UCC1 and UCC3 are supported on the device.

The following features are not supported: USB, MCC, SPI2, Ethernet, and ATM.

On this device, only a system bus is supported; references to a secondary bus should be disregarded. As such,

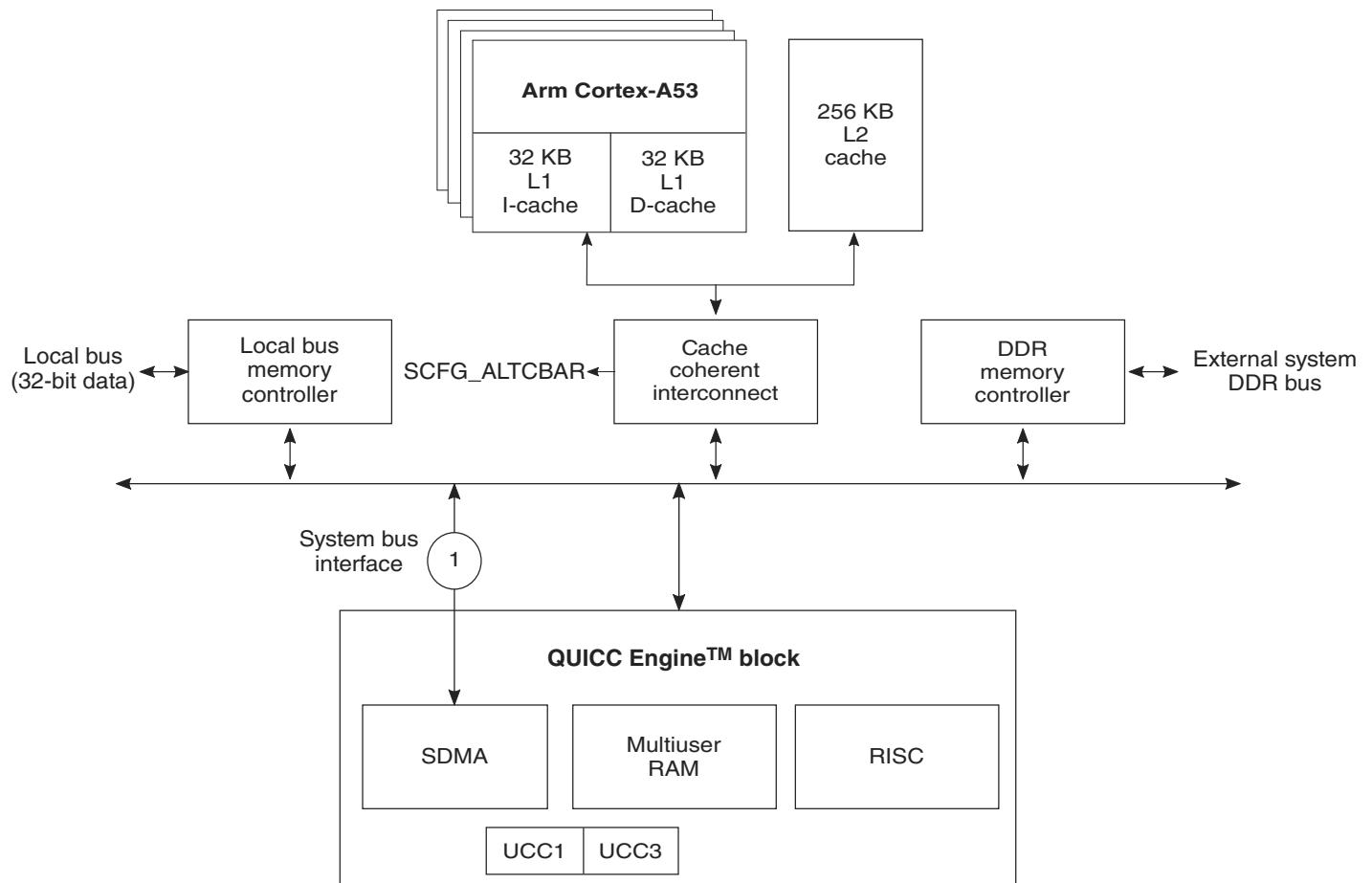
- In the "Bus Selection Mechanisms" subsection, references to the secondary bus should be disregarded.
- In the "Serial DMA Status Register (SDSR)" subsection, the BER\_2 bit is not available and should be reserved.

- In the "Serial DMA Mode Register (SDMR)" subsection, BER\_2\_MSK and SBER\_2\_MSK bits are not available and should be reserved.
- In the "Serial DMA Transfer Address Registers (SDTA1 and SDTA2)" and "Serial DMA Transfer Communication Channel Number Registers (SDTM1 and SDTM2)" subsections, references to the secondary bus should be disregarded.

### 15.3.1.1 System interface-data paths

In the "Data paths" subsection of the QEIWRM, use this Arm Cortex core information:

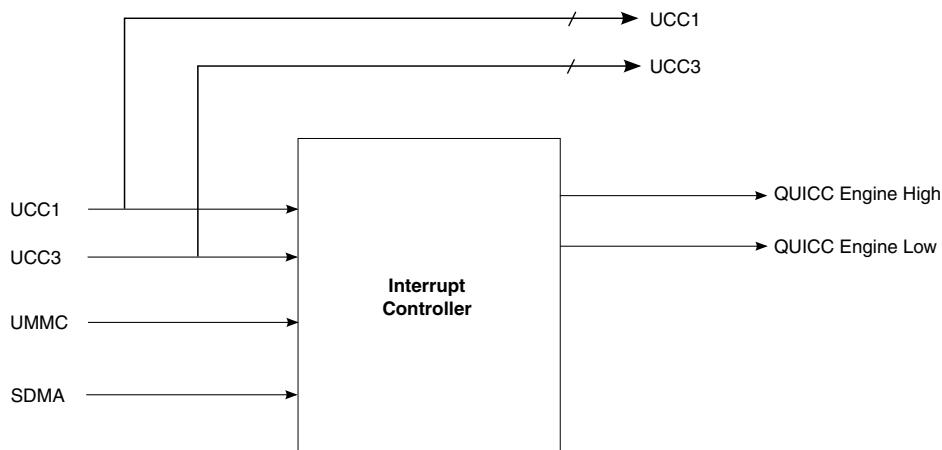
The figure below is a simplified block diagram that shows the data paths. The data paths include a path from the SDMA system bus interface to targets connected to the Cache coherent interconnect. block shown in the figure, is responsible for distribution of I/O masters transactions to the various possible targets (such as the PCI Express or DDR memory controller) based on the transaction's address. The SDMA channel must be configured for big-endian byte ordering for accessing buffer data. The big-endian byte ordering format is programmed in the receive and transmit bus mode registers associated with the peripherals (UMCC). Refer to each protocol of these peripherals for programming of this feature.



**Figure 15-2. Data paths**

### 15.3.1.2 System interface-interrupt configuration

In the "Interrupt Configuration" subsection, use the figure below to show the QUICC Engine interrupt structure.



**Figure 15-3. QUICC engine module interrupt structure**

### 15.3.1.3 System interface-bus arbitration

The "Bus Arbitration" subsection should be disregarded.

## 15.3.2 Configuration

Of the UCCs, this device only supports UCC1 and UCC3.

The following features are not supported: USB, MCC, SPI2, Ethernet, and ATM.

### 15.3.2.1 Configuration-parameter RAM

For the device, replace the "Default Parameter RAM Base Addresses" table with the table below.

**Table 15-2. Default parameter RAM base addresses**

Address	Peripheral	Size (Bytes)
0x3400	UCC1 (Rx and Tx)	256
0x3600	UCC3 (Rx and Tx)	256
0x3A00	TIMER	64

### 15.3.3 Multiplexing and timers

Of the UCCs, only UCC1 and UCC3 are supported; of the TDMs, only TDMA and TDMB are supported.

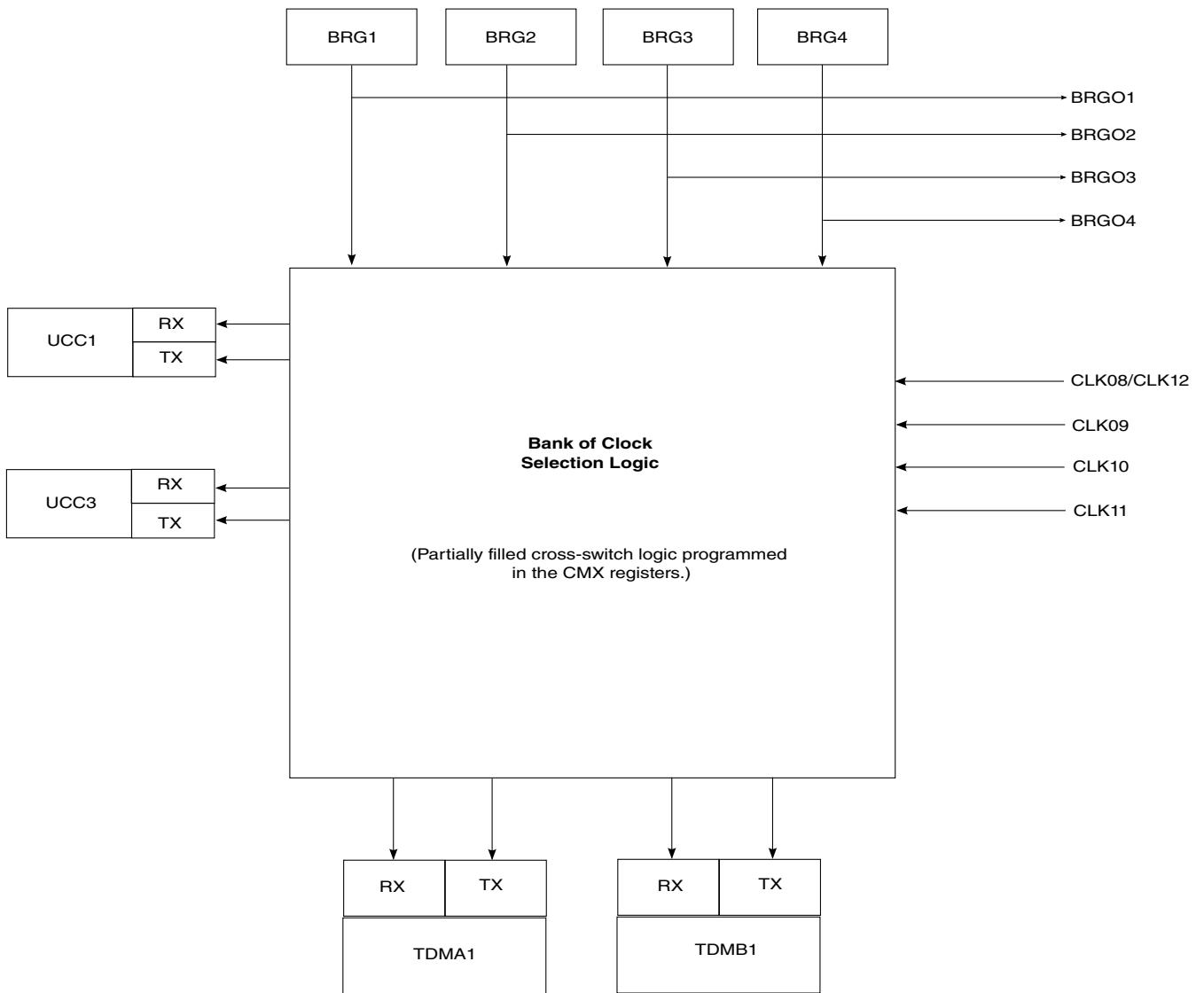
USB, SPI, Ethernet, and ATM are not supported.

The following information applies:

- A bank of 5 external clocks CLK[8-12]
- Two internal clocks CLK[3,15]
- Four supported BRGs: BRG[1-4]
- Two UCCs: UCC1 and UCC3
- Two TDMs: TDMA and TDMB

#### 15.3.3.1 Multiplexing and timers-NMSI configuration

For the device, replace the "Bank of Clocks" figure with the figure below.

**Figure 15-4. Bank of clocks****NOTE**

1. CLK3 and CLK15 are connected in chip level to the internal platform clock.
2. External CLK12 pin is connected internally to both CLK12 and CLK8 pins of QE.

In the "NMSI Configuration" subsection, refer to the tables below.

The table below shows the clock source options for the serial controllers and TDM channels.

**Table 15-3. Clock source options-external clock signals**

Clock	External CLK Number						
	3	8	9	10	11	12	15
UCC1 Rx			V	V	V	V	V
UCC1 Tx			V	V	V	V	V
UCC3 Rx			V	V	V	V	V
UCC3 Tx			V	V	V	V	V
TDMA1 Rx	V	V					
TDMA1 Tx			V				
TDMB1 Rx				V			
TDMB1 Tx					V		
QUICC Engine Timer					V	V	

The table below shows the clock routing options using the internal clock generators.

**Table 15-4. Clock source options-internal clock**

Clock	BRG clock number			
	1	2	3	4
UCC1 Rx	V	V		
UCC1 Tx	V	V		
UCC3 Rx	V	V		
UCC3 Tx	V	V		
TDMA1 Rx <sup>1</sup>			V	V
TDMA1 Tx <sup>1</sup>			V	V
TDMB1 Rx <sup>1</sup>			V	V
TDMB1 Tx <sup>1</sup>			V	V

1. TDM options are used only for internal loopback mode.

The table below shows the UART autobaud clock options.

**Table 15-5. Clock source options-UART autobaud clock options**

Clock	BRG Clock Number for UART Autobaud			
	1	2	3	4
UCC1 Rx	V			
UCC1 Tx	V			
UCC3 Rx		V		
UCC3 Tx		V		

### 15.3.3.2 Multiplexing and timers-baud-rate generators (BRGs), BRG configuration registers 1-4 (BRGCn)

The table below describes the BRGCn fields.

**Table 15-6. BRGCn field descriptions**

Bits	Name	Description
0-13	-	Reserved, should be cleared.
14	RST	<p>Reset BRG. Performs a software reset of the BRG identical to that of an external reset. A reset disables the BRG and drives BRGO high. This is externally visible only if BRGO is connected to the corresponding parallel I/O pin.</p> <p>0 Enable the BRG. 1 Reset the BRG (software reset).</p>
15	EN	<p>Enable BRG count. Used to dynamically stop the BRG from counting-useful for low-power modes.</p> <p>0 Stop all clocks to the BRG. 1 Enable clocks to the BRG.</p>
16-17	EXTC	<p>External clock source. Selects the BRG input clock.</p> <p>00 The BRG input clock comes from the BRGCLK (internal clock generated from the QUICC Engine clock, it is one-half of the QUICC Engine clock). 01 If BRG1, 2: The BRG input clock comes from the CLK3 pin (driven internally by platform/2 clock). If BRG3, 4: The BRG input clock comes from the CLK9 pin. 10 If BRG1, 2: The BRG input clock comes from the CLK3 pin (driven internally by platform/2 clock). If BRG3, 4: The BRG input clock comes from the CLK15 pin (driven internally by platform/2 clock). 11 Reserved</p>
18	ATB	<p>Autobaud. Selects autobaud operation of the BRG on the corresponding RXD. ATB must remain zero until the UCC receives the three Rx clocks. Then the user must set ATB to obtain the correct baud rate. After the baud rate is obtained and locked, it is indicated by setting AB in the UART event register, see "UCC UART Event Register (UCCE) and Mask Register (UCCM)," in <i>QUICC Engine™ Block Reference Manual with Protocol Interworking</i>.</p> <p>0 Normal operation of the BRG. 1 When RXD goes low, the BRG determines the length of the start bit and synchronizes the BRG to the actual baud rate.</p>
19-30	CD	<p>Clock divider. CD presets an internal 12-bit counter that is decremented at the DIV16 output rate. When the counter reaches zero, it is reloaded with CD. CD = 0xFFFF produces the minimum clock rate for BGRO (divide by 4,096); CD = 0x000 produces the maximum rate (divide by 1). When dividing by an odd number, the counter ensures a 50% duty cycle by asserting the terminal count; once on clock low and next on clock high. The terminal count signals that the counter has expired and then toggles the clock.</p> <p>0x000 Divide by 1 (maximum clock rate) 0x001 Divide by 2 ... 0xFFFF Divide by 4096 (minimum clock rate)</p>
31	DIV16	<p>Divide-by-16. Selects a divide-by-1 or divide-by-16 prescaler before reaching the clock divider.</p> <p>0 Divide by 1. 1 Divide by 16.</p>

The table below shows the possible external clock sources for the BRGs.

**Table 15-7. BRG external clock source options**

BRG	CLK						
	3 1	8	9	10	11	12	15 1
BRG1	V						
BRG2	V						
BRG3			V				V
BRG4			V				V

1. CLK3 and CLK15 are connected to the internal platform/2 clock.

## 15.3.4 Unified communications controllers (UCCs)

Of the UCCs, only UCC1 and UCC3 are supported.

The following features are not supported: USB, MCC, SPI, Ethernet, and ATM.

### 15.3.4.1 Unified communications controllers (UCCs)-UCC page base address

Refer to the table below for the UCC base addresses.

**Table 15-8. Parameter RAM-UCC default base addresses**

Page	Address <sup>1</sup>	Peripheral	Size (Bytes)
1	0x3400	UCC1	256
3	0x3600	UCC3	256

1. Offset from RAM\_Base

## 15.3.5 HDLC controller

The ratio between the HDLC interface serial clock frequency and the QUICC Engine clock frequency should be at least 1:3.

### NOTE

The HDLC serial clock must not exceed the maximal frequency as specified in the data sheet.

### 15.3.5.1 HDLC controller-introduction

It should be noted that GUMR[RTSM] should be set for the HDLC nibble mode.

### 15.3.6 Transparent controller

The chip does not support the automatic sync feature, so all references to this should be disregarded.

The QUICC Engine frequency must be higher than the data bit-rate multiplied by 8/3.

The transparent protocol is supported on all UCCs, but octal mode is not supported. UCC1 only supports 1-bit data (serial) mode, while UCC3 supports 1- and 4-bit data modes.

### 15.3.7 UCC for fast protocols

Only a system bus is supported; references to a secondary bus should be disregarded.

As such,

- In the "Bus Mode Registers (RBMR and TBMR)" subsection of the QEIWRM, DTB and BDB should be reserved.

### 15.3.8 UCC for slow protocols

Only a system bus is supported; references to a secondary bus should be disregarded.

As such,

- In the "Bus Mode Registers (RBMR and TBMR)" subsection of the QEIWRM, DTB and BDB should be reserved.

### 15.3.9 BISYNC mode

The ratio between the BISYNC serial clock frequency and the QUICC Engine clock frequency should be at least 1:7.

Generally, the BISYNC serial frequency is less than 20 MHz.

### 15.3.10 Serial interface with time-slot assigner

The two TDMs, TDMA and TDMB, only support 1-bit data (serial) mode.

The TDM can only work with an external sync and external clock; other options are not supported. The external clock and external sync need to meet the timing specifications as provided in data sheet.

This device supports TDM in a high-speed mode. To run in this mode, the QUICC Engine platform to TDM interface frequency ratio should be 8:1, where a ratio of 400 MHz:50 MHz is recommended. The TDM maximum frequency is 50 MHz.

For the ISDN protocol, the maximum frequency supported is 25 MHz, with the condition that the QUICC Engine platform to ISDN interface frequency ratio is 16:1. For example, if the QUICC Engine platform frequency is 400 MHz, then the ISDN interface maximum frequency is 25 MHz.

### 15.3.11 Multi-channel controller on UCC (UMCC)

UMCC is supported with only the HDLC and transparent modes. The SS7 mode is not supported.

- The UMCC controller supports two TDM lines.

# Chapter 16

## Data Path Acceleration Architecture (DPAA)

### Overview and SoC DPAA Implementation

#### 16.1 DPAA Introduction and Terms

The data path processing subsystem is an implementation of the QorIQ Data Path Acceleration Architecture (DPAA).

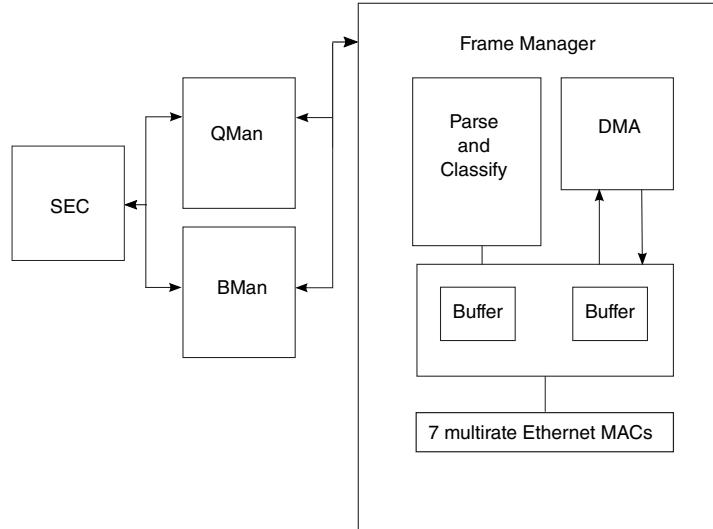
*The QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* describes the superset of DPAA functionality. The chip implements a unique subset of this functionality, as described in the [LS1043A-Specific DPAA Implementation Details](#).

As shown in the following figure, the DPAA consists of a parse, classify, and distribute module (PCD) and an accelerator module (FMan), along with these related network interfaces:

- Multirate Ethernet MAC (mEMAC)
- hardware offload accelerators:
  - SEC
  - the infrastructure blocks that support these accelerators (QMan and BMan)

Each of these accelerators provides its own unique set of functionality, and thus each has its own unique requirements for how software must program and interact with that accelerator.

The DPAA's common infrastructure modules and the desire to be able to pass data received from one accelerator to another led to some common requirements, restrictions, and conventions. This chapter outlines requirements and conventions that are common to multiple accelerators within the DPAA.



**Figure 16-1. QorIQ Data Path Acceleration Architecture**

The following list describes common DPAA terms and definitions.

#### Buffer

Region of contiguous memory, allocated by software, may be managed by the DPAA Buffer Manager (BMan).



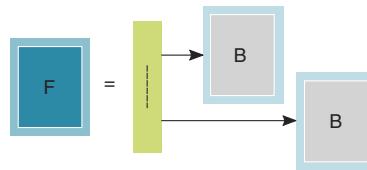
#### Buffer pool

Set of buffers with common characteristics (mainly size, alignment, access control)



#### Frame

Contents of a single buffer or multiple buffers referenced by a table that hold data. For example, packet payload, header, and other control information.



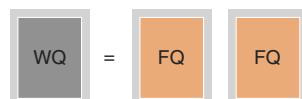
#### Frame queue (FQ)

FIFO of frames



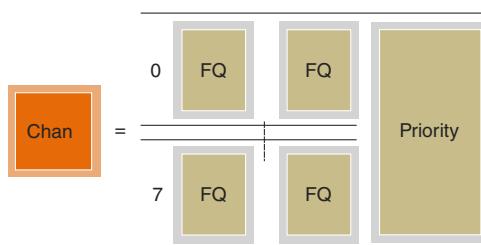
#### Work queue (WQ)

FIFO of frame queues (FQs)



#### Channel

Set of eight work queues (WQs), with hardware provided prioritized access



#### Dedicated channel

Channel statically assigned to a particular end point from which that end point can dequeue frames. End-point may be a CPU, FMan, or SEC.

#### Pool channel

A channel statically assigned to a group of end points from which any of the end points may dequeue frames.

## 16.2 Data Formats Used in the DPAA

Data-to-be-processed is passed within the DPAA in distinct units known as “frames.”

The actual data is held in buffers in memory and a description of the frame is what is passed within the DPAA.

## 16.2.1 Frame Descriptor (FD)

A frame descriptor (FD) is the basic element queued by the QMan.

See the, "Frame Manager" chapter in the *QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* for a description of the QMan's usage of the FD. The use and interpretation of some fields is specific to the producer or consumer of the FD. Such usage is described in the chapter/section covering that module.

### 16.2.1.1 FD Format

**Table 16-1. Frame Descriptor (FD)**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
DD	ICID							BPID							Reser	ve	EICID	Reserved			ADDR										
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
ADDR																															
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
FORMAT	OFFSET							LENGTH																							
	LENGTH or CONGESTION WEIGHT																														
96	97	98	99	10	10	10	10	10	10	10	10	10	10	10	10	11	11	11	11	11	11	11	11	12	12	12	12	12	12	12	12
				0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
STATUS/CMD																															

**Table 16-2. FD Field Descriptions**

FD Bits	FD Name	FD Field Description
0-1	DD	Dynamic Debug marking code point A frame with a non-zero debug code point can generate a trace event at various enqueue and/or dequeue points within the QMan. Individual QMan users have different mechanisms and criteria for setting non-zero DD values.
2-7	ICID	Frame ICID (6 least significant bits of complete ICID) The ICID is set when the FD is enqueued. It is used by a hardware module that dequeues the FD as part of the memory access control mechanism supported by the SMMU.
8-15	BPID	Buffer Pool ID The Buffer Pool ID is the number of the BMan buffer pool to which the buffer at ADDR should be released, if it is to be released to BMan.
16-17	-	Reserved
18-19	EICID	Frame Extended ICID (2 most significant bits of complete ICID) The complete 8-bit ICID value is a concatenation of {EICID, ICID}.

Table continues on the next page...

**Table 16-2. FD Field Descriptions (continued)**

FD Bits		FD Name	FD Field Description
20-23		-	Reserved
24-63		ADDR	<p>Address</p> <p>The memory address of the start of the buffer holding the frame data or the buffer containing the scatter/gather list describing the frame</p>
64-66		FORMAT	A coded value that defines the format of the OFFSET and LENGTH fields, and of the frame referenced by this FD. See <a href="#">Frame Format Codes</a> for a description of these codes.
Optional fields	67-75	OFFSET	<p>This field is valid only when FORMAT bits 65-66 are 00; otherwise, a frame offset value of 0 is implied.</p> <p>Frame offset is the number of bytes from the frame address (ADDR) at which actual data begins. Note that this field is not present in all FDs</p>
	76-95	LENGTH	Total number of valid bytes of data in the frame. Note that this field is not present in all FDs.
	67-95	CONGESTION WEIGHT	A value used by the QMan for certain congestion management/avoidance calculations. Note that this field is not present in all FDs.
96-127		STATUS/CMD	<p>Status or Command</p> <p>This field allows the sender of a frame to communicate some out-of-band information to the receiver of the frame. For example, this field can be used to communicate a command describing what processing is to be performed to a hardware accelerator or the error/output status after a hardware accelerator has finished processing a frame.</p>

## 16.2.2 Multi-Buffer Frames

The actual data in a frame is held in one or more memory buffers.

If multiple buffers are required, a scatter/gather table (also known as a block vector or IO vector table) is used to describe the buffers in a multi-buffer frame. See [Scatter/Gather Entry Format](#) for a description of this data structure.

### 16.2.2.1 Scatter/Gather Entry Format

The table below lists the Scatter/Gather Table Entry Format:

**Table 16-3. Scatter/Gather Table Entry Format**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	Reserved (must be 0)																												ADDR			
	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
1	ADDR																															
	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
2	E	F	LENGTH																													

*Table continues on the next page...*

**Table 16-3. Scatter/Gather Table Entry Format (continued)**

	96	97	98	99	10	10	10	10	10	10	10	10	10	10	11	11	11	11	11	11	11	11	12	12	12	12	12	12	12
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	
3	Reserved (must be 0)												BPID		Reserved (Must be 0)	OFFSET													

**Table 16-4. Scatter Gather Table Entry Field Description**

Bits	Name	Description
0-23	-	Reserved (must be 0)
24-63	ADDR	Address of the buffer referenced by this table entry. This buffer may contain data or it may contain another scatter/gather table.
64	E	Extension bit. If set, this table entry points to a buffer that contains another scatter/gather table.
65	F	Final bit. If set, this is the last table entry for this frame.  Note that the E (extension) bit takes precedence over the F (final) bit. If both are set, the F bit is ignored and processing continues with the entries in the scatter/gather table in referenced buffer.
66-95	LENGTH	Depending on the context, this field either specifies the number of valid bytes of data in the referenced buffer or the size of the referenced buffer (in the case that empty buffers are being provided to an accelerator).  Note that if the E (extension) bit is set, LENGTH is ignored.
96-103	-	Reserved (must be 0)
104-111	BPID	Buffer Pool ID: The number of the BMan buffer pool to which this buffer should be released, if it is released to BMan.
112-114	-	Reserved (must be 0)
115-127	OFFSET	An offset in bytes from the ADDR at which valid data starts  Note that if the E (extension) bit is set, "valid data" is a scatter/gather table.

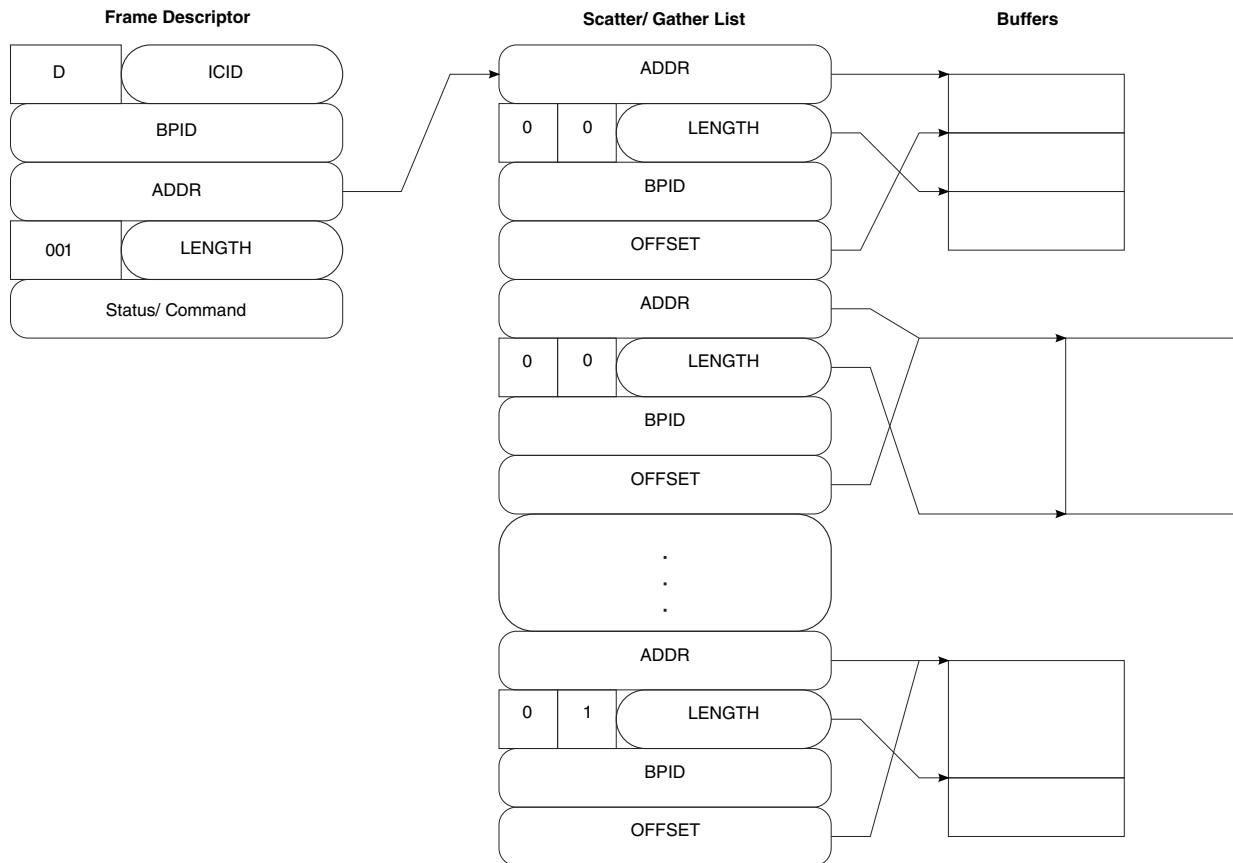
### 16.2.2.2 Multi-Buffer Frame Considerations

Important multi-buffer frame considerations are as follows:

- The first scatter/gather table in a multi-buffer frame is held in the buffer referenced by the FD. In the case of a short, multi-buffer frame, the table starts at OFFSET bytes from the beginning of the buffer.
- “Trees” or hierarchy are not supported, because for simple, multi-buffer frames, processing never returns to the table in the original buffer.
- If the E bit is set, it indicates that the table entry points to another buffer containing more table entries. When a scatter/gather table entry with the E bit set is encountered, processing proceeds from the first entry in the new table found in the new buffer. This new table may begin at some offset from the start of the buffer as defined by the OFFSET field in the entry with the E bit set.

- In simple, multi-buffer frames, the LENGTH field in a table entry with its E bit set can be ignored.
- The E bit takes precedence over the F bit (that is, if both are set in an entry, the F bit is ignored).

This figure shows a simplified representation of a "long" simple frame using a scatter/gather table. Note that this diagram does not show the use of the Extension bit.



**Figure 16-2. Simplified Representation of a Long, Multi-Buffer, Simple Frame**

### 16.2.2.3 Situations Where Multi-Frame Buffering Stops

Multi-buffer frame processing stops in the following situations:

- When the data referenced by an entry with the F bit is processed regardless of the LENGTH indicated in the FD for the frame. In some cases, a mismatch (less data found in the frame compared to the FD length) may be an error condition for the accelerator module that is performing the processing, and it is reported as such.
- Processing also stops when the number of bytes specified by the overall length in the frame's FD have been processed. In this case, it is not necessary to have encountered an entry with the F bit set and it is not considered an error when this occurs.

### 16.2.3 Single-Buffer Frames

Most frame formats consist of a single delineated unit of data such as a single packet or Protocol Data Unit (PDU).

For a single buffer, the FD[LENGTH] field represents the total amount of valid data in the frame. The OFFSET represents the offset to the start of valid data.

For multiple buffers (used for scatter/gather), the FD[LENGTH] field still represents the total amount of valid data in the frame, but the OFFSET represents the offset to the start of the scatter/gather table in the first buffer.

### 16.2.4 Compound Frames

Frames may also consist of multiple, related, distinct units such as the encrypted form of a packet along with the decrypted form of the same packet. These are known as compound frames, which consist of two or more simple frames.

Each simple frame in a compound frame can in turn be stored in a single buffer or in multiple buffers. Compound frames exist so that all of the related data can be passed in a single unit within the DPAA. If an FD has a FORMAT code that identifies it as a compound frame then the FD[CONGESTION WEIGHT] is used to by the QMan for active queue management calculations such as WRED.

#### 16.2.4.1 When to Use Compound Frames

Compound frames are used for the following purposes:

- To pass multiple, distinct pieces of data either to or from a hardware accelerator. For instance, it may be required to pass frames containing both encrypted and decrypted forms of a packet.
- To supply empty buffers to a hardware accelerator into which it may place its output. In this case, it is not desirable to use the BMan. Any LENGTH and OFFSET fields

that refer to a specific buffer have special meaning; OFFSET specifies the byte offset from ADDR where the accelerator should start storing data and LENGTH gives the number of bytes of data which can be stored in the remainder of the buffer.

Consider the following when using compound frames:

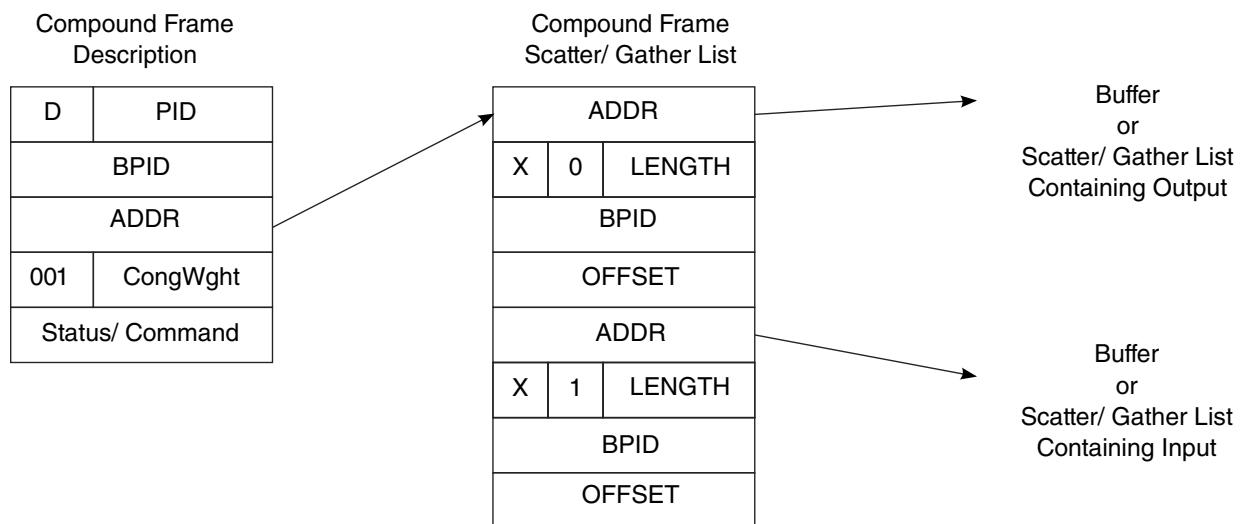
- When multiple input or output frames are described by a compound frame, their order is consumer-dependent.
- If a compound frame is used to pass empty buffers to a consumer for its output, those buffers are in the first frame of the compound frame.
- If a module uses a compound frame to return the input frame as well as its output frame, the output frame is the first frame of the compound frame and the input frame is the second frame.

#### 16.2.4.2 Compound Frame Considerations

Important compound frame considerations are as follows:

- Compound frames use the same scatter/gather table format as simple frames, but with slightly different semantics for the fields in the table entries.
- The buffer referenced by FD[ADDR] of a compound frame must contain a scatter/gather table (the compound scatter/gather table). A compound frame contains FD[CONGESTION WEIGHT] but does not contain FD[LENGTH] or FD[OFFSET]. The compound scatter/gather table must start at FD[ADDR].
- Each entry in the compound scatter/gather table references a simple frame. ADDR, LENGTH, BPID, and OFFSET fields in the compound scatter/gather table entry replace the corresponding fields in an FD. A compound frame contains FD[CONGESTION WEIGHT] but does not contain FD[LENGTH] or FD[OFFSET]. The compound scatter/gather table must start at FD[ADDR].
- The E bit is set if the simple frame occupies multiple buffers. The E bits in multiple entries in the compound scatter/gather table may be set, because each simple frame in the compound frame may occupy multiple buffers. In this case, the buffer at ADDR contains a scatter/gather table.
- The scatter/gather table(s) that make up a simple frame that is part of a compound frame follow the same semantics as described in [Scatter/Gather Entry Format](#).
- There is no equivalent in the compound scatter/gather table for the FORMAT field in the FD. The entries in a compound scatter/gather table cannot themselves be compound frames; the E bit is used to indicate a multi-buffer frame, and the LENGTH and OFFSET fields supported in the scatter/gather entry are a superset of the fields in an FD.
- The F bit must be set in the last entry of a compound frame scatter/gather table.

This figure shows a representation of a compound frame.



**Figure 16-3. Simplified Representation of a Compound Frame**

## 16.2.5 Simple Frames

Simple frames can be either short or long. Regardless of the length of the frame, data is stored in a single-buffer frame, whereas scatter/gather tables are stored in a multi-buffer frame.

A short frame can be no more than (1 Mbyte - 1 byte)-long but the data (in the case of a single buffer frame) or scatter/gather table (in the case of a multi-buffer frame) can be stored starting at an offset from the beginning of the buffer referenced by the FD. Long frames can be as much as (512 Mbytes - 1 byte)-long, but the data or scatter/gather table must be stored starting at the beginning of the buffer referenced by the FD.

Because network data (packets) are typically much less than 64 Kbytes in size, the short format is useful for most network processing needs.

## 16.2.6 Frame Format Codes

This table defines the frame format codes and provides a brief description of the frame format.

**Table 16-5. Frame Format Codes**

Value	Mnemonic	Definition	Size of LENGTH, OFFSET, and CONGESTION WEIGHT Fields
'000'	Short single buffer simple frame	Simple frame Single buffer, Offset and "small" length	9b OFFSET, 20b LENGTH
'010'	Long single buffer simple frame	Simple frame, single buffer, "big" length	29b LENGTH No OFFSET
'100'	Short multi-buffer simple frame	Simple frame, Scatter Gather table, Offset and "small" length	9b OFFSET, 20b LENGTH
'110'	Long multi-buffer simple frame	Simple frame, Scatter Gather table, "big" length	29b LENGTH No OFFSET
'001'	Compound frame	Compound Frame	29b <CONGESTION WEIGHT> No LENGTH or OFFSET
'011'	NA	Reserved	Not defined
'101'	NA	Reserved	Not defined
'111'	NA	Reserved	Not defined

## 16.2.7 Frame Formats Supported by Accelerators

Hardware accelerators (FMan) support a subset of the DPAA frame formats.

This table summarizes the frame formats supported by various accelerators.

**Table 16-6. Frame Format Support Matrix**

	Short, Single Buffer		Long, Single Buffer		Short Multi-Buffer		Long Multi-Buffer		Compound Frames	
	Input	Output	Input	Output	Input	Output	Input	Output	Input	Output
FMan	Yes	Yes	No	No	SG table <sup>1</sup> limited to 16 entries. E bit not supported	SG table limited to 16 entries E bit not supported	No	No	No	No
SEC	Yes	Yes	Yes	Yes	Yes	SG table limited to one buffer E bit not supported	Yes	SG table limited to one buffer E bit not supported	Yes	Yes

1. Scatter/Gather entry table

## 16.2.8 Special Values and Exceptions

The ADDR, BPID, and LENGTH fields can have special meanings under certain conditions, as follows:

- FDs
- Compound frame scatter/gather tables
- Multi-buffer scatter/gather tables

Exceptions and special behaviors are as follows:

- When the LENGTH field that describes a specific buffer is zero, this indicates that the buffer contains no data. If an accelerator finds a LENGTH of zero it should not access (read or write) the memory at ADDR.
- The LENGTH field in a scatter/gather table entry of a simple multi-buffer frame with its E (extension) bit set is an exception, because the value of this field is ignored.
- FDs, compound frame, or multi-buffer frame SG entries with ADDR/BPID/LENGTH=0 encoding indicate that the FD or SG entry does not convey buffer information, that is, the SG entry is unused and is therefore skipped during input and/or output processing.

## 16.2.9 Releasing Buffers to the BMan

Accelerators only release buffers to the BMan that they have processed or, in the case of a buffer with a zero LENGTH, that they have passed over during processing. Buffers with a zero ADDR, BPID, and LENGTH are not released.

**REQUIREMENT:** Because processing stops when the frame's total length of data is processed or after the data referenced by a scatter/gather entry with its F (final) bit set is processed, ensure that all buffers in a frame are released. For example, software must ensure that there are no entries in a scatter/gather table for a multi-buffer frame that are beyond the frame's total length or described in entries after the entry with the F bit set.

## 16.3 Accessing Memory Using Isolation Context Identifier(ICID)

The isolation context identifier(ICID) maps an incoming transaction from IO device to one of the context and it maps to StreamID as described in Arm documentation.

By using different ICIDs, a single hardware module can perform memory accesses on behalf of different requestors with the mapping and access controls appropriate to that requestor.

## ICID Requirements

Some important ICID requirements are as follows:

- The requestor must communicate the ICID to the hardware blockmodule. In the DPAA, this is done using the ICID. And, this is formed by concatenating ICID and EICID from the FD with the latter used as the 2 most significant bits(msbs).
- Because ICID is in the FD, the module can use an ICID for each frame to access the memory.
- Some hardware module may make different types of memory accesses as a result of dequeuing a frame. For instance, they may access per queue context/state descriptors as well as reading and writing frame data. These modules use the same ICID from FD for these different types of access.
- The ICID must be set by the hypervisor. For software portals, this is done by the QMan from values configured for the portal to ensure that accesses by hardware module on behalf of software are controlled. In other words, software running on a core cannot get a hardware blockmodule to make accesses to memory, because it is not permitted to make these accesses by setting the ICID value in an FD.

## 16.4 Packet Walk-Through Example

The following example walks a packet through a DPAA-enabled, QorIQ device to illustrate how the DPAA offloads and accelerates packet forwarding while leaving key decisions under software control:

1. Datapath processing begins when Ethernet frames arrive at a network interface, assumed to be one of the Ethernet MACs within a Frame Manager (FMan). Alternatively, packets can arrive across a peripheral bus from an external network interface, in which case, the same packet walk-through stages can be applied with the help of a CPU acting as an I/O processor.
2. After FMan receives the Ethernet frame, it requests one or more buffers from the hardware Buffer Manager (BMan) to store the frame. BMan maintains pools of buffers, each with software-defined characteristics, and FMan is initialized to request a buffer from the most appropriate pool. If a sufficiently large buffer cannot be found for the incoming frame, FMan stores the frame across several smaller buffers and creates a scatter/gather list for these buffers.
3. FMan's configurable parsing and filing capabilities can perform initial classification, sufficient to steer a packet toward a control processor, or toward one of the datapath

processors for flow-specific processing (or additional classification by means of software). The steering of a packet towards a processor or a group of processors could be also based on differentiating flows by means of the Quality of Service attributes of the flow (for example, DSCP, IP precedence, or by user-defined proprietary headers).

4. Steering is accomplished by FMan issuing an enqueue command to QMan with the designated Frame Queue ID, along with frame parameters such as Frame Queue ID and Color Marking. In this example FMan's initial classification causes it to select a Frame Queue ID which the Queue Manager uses to steer the Frame Queue to the dedicated channel of logical CPU#3, a CPU operating in a datapath role. Potentially, the Frame Queue could be selected based on the QoS requirements of the flow with a policing profile associated with the selection.
5. Continuing the example, QMan places the Frame Queue onto Work Queue#5 of CPU#3's dedicated channel. The Frame Queue was queued to CPU#3 due to user configuration decisions that all packets belonging to this specific flow should be processed by CPU#3, possibly to take advantage of special processing instructions locked in CPU#3's private cache, or due to user-defined load balancing. The Frame Queue was placed in Work Queue#5 for QoS reasons, as the amount of traffic processed from each Work Queue relative to other work queues is also user-configurable. All Frame Queues on a Work Queue have equal priority, but the amount of traffic that the CPU draws from each is user-configurable to allow fairness and appropriate bandwidth allocation.
6. Once configured, QMan can take care of the packet/frame level scheduling requirements of the CPU, that is, QMan would appropriately schedule the Work Queue and Frame Queue within the Work Queue. QMan can be configured to perform a stashing operation in response to a CPU (or co-processor) accessing a Frame Queue.
7. CPU#3 performs protocol processing on the packet, including modification of the packet data in the buffer. Any modifications which change the length of the packet would be noted by means of changes to the frame. The CPU determines that the packet requires IPsec ESP processing, and enqueues the frame back to the QMan using the FQ ID associated with the packet's specific ESP tunnel.
8. The QMan uses this Frame Queue ID to determine that the next consumer of the Frame Queue is the SEC, and places the Frame Queue onto Work Queue#5 of the SEC's dedicated channel. The SEC pulls Frame Queues from the Work Queues in its dedicated channel according to user-defined weights in a WFQ model.
9. The SEC dequeues and processes data from the Frame Queue, adding the tunnel IP header, ESP header, IV, trailer, and HMAC, and writing encrypted data to either the original buffers, or new buffers which the SEC requests from BMan. The SEC updates the frame description (length change due to the addition of the headers, trailers, and HMAC) and enqueues the FQ back to QMan using a configured FQ ID.

In this case, the FQ ID causes the QMan to enqueue the FQ to Work Queue #5 of logical CPU#4's dedicated channel. CPU#4 dequeues the FQ, determines the outbound interface for the newly encrypted packet, updates the tunnel IP header, and enqueues the frame back to QMan on a new FQ ID. This FQ ID causes the QMan to enqueue the FQ to a channel serviced by FMan, which dequeues the FQ, transmits one or more packets from that FQ out the appropriate mEMAC, and releases the buffers back to BMan.

The processing pipeline used in this example is not required by the QorIQ DPAA. The initial classification could have caused the packet to be steered toward a CPU dedicated to fine-grained classification, or to a pool channel of CPUs, any of which could have performed the operations described. CPU#3 could have added the ESP header and trailer to the packet and sent it to the SEC for crypto-only processing. Following SEC processing, the flow was steered to CPU#4, however it could have just as easily been steered back to CPU#3, or to a pool channel. At any FQ ID transition, the relative priority of the flow could have been elevated or reduced by enqueueing it to a different work queue.

## 16.5 LS1043A-Specific DPAA Implementation Details

The *QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* describes the superset of DPAA functionality. The LS1043A implements a unique subset of this functionality, as described in the following sections.

### 16.5.1 Queue Manager (QMan) Implementation

The QMan of the LS1043A is implemented as described in the *QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* with the following implementation parameters:

- QMan block base address: 188\_0000h
- 512 frame queue (FQ) cache
- 2-KB SFDRs
- 256 congestion groups

#### NOTE

The QMan is always a non-secure master. The security protection (CSL6[24:16]) in the QMan/BMan portal path should be configured as allow all access.

## 16.5.2 Buffer Manager (BMan) Implementation

The BMan of the LS1043A is implemented as described in the *QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* with the following implementation parameters:

- BMan block base address: 189\_0000h
- 64 buffer pools

## 16.5.3 Frame Manager (FMan) Implementation

The FMan is implemented as described in the *QorIQ LS1043A Data Path Acceleration Architecture (DPAA) Reference Manual* with the following implementation parameters:

- FMan block base address: 1A0\_0000h
- Seven multirate Ethernet MACs
  - Five 1G/100M/10M multirate Ethernet MACs
  - One 2.5G/1G/100M/10M multirate Ethernet MAC
  - One 10G/2.5G/100M/10M multirate Ethernet MAC
- Block base addresses are as follows:
  - FM1 mEMAC1: 1AE\_0000h
  - FM1 mEMAC2: 1AE\_2000h
  - FM1 mEMAC3: 1AE\_4000h
  - FM1 mEMAC4: 1AE\_6000h
  - FM1 mEMAC5: 1AE\_8000h
  - FM1 mEMAC6: 1AE\_A000h
  - FM1 mEMAC9: 1AF\_0000h
- Supports 1 host command and 3 offline ports:
  - Host command: 02h
  - Offline port 3: 03h
  - Offline port 4: 04h
  - Offline port 5: 05h
- FM1 Dedicated MDIO1: 0x1AF\_C000
- FM1 Dedicated MDIO2: 0x1AF\_D000
- One FMan Controller complex
- 384-KB internal FMan memory
- 64-KB FMan Controller configuration data
- Up to 32 Keygen schemes
- Up to 256 Policer profiles
- Up to 84 entries in FMan DMA command queue

- Up to 64 TNUMs
- Up to 1 FMan debug flows

#### 16.5.4 Security and Encryption Engine (SEC) Implementation

The Security and Encryption Engine of the LS1043A is implemented as described in the *QorIQ LS1043A Security (SEC) Reference Manual* with the following implementation parameters:

- SEC block base address: 170\_0000
- 5 Gbps IPsec performance at Ethernet MTU-sized packets, when the cipher suite is AES-128-CBC / HMAC-SHA-1
- Cryptographic Hardware Accelerators (CHAs) include:
  - PKHA
  - DESA
  - AESA
  - MDHA
  - RNG4
  - CRCA
  - SNOWf8 and SNOWf9 (Snow)
  - KFHA (Kasumi)
  - ZUCA and ZUCE

#### NOTE

For read transactions to be coherent from SEC, both of the registers should be configured:

- SCFG\_SNPCNFGCR[SECRDSNP] (controls the AxDomain port-connectivity)
- MCFGR[ARCACHE] (controls the AxCache port-connectivity. For more information on MCFGR register, refer security reference manual.)



# Chapter 17

## Secure Boot and Trust Architecture

### 17.1 Trust architecture objectives

The LS1043A supports the QorIQ Trust Architecture 2.1.

The objective of the trust architecture is to allow OEMs to prevent or strongly mitigate an attacker's ability to achieve the following attacks:

- theft of functionality,
- theft of third-party data,
- and theft of uniqueness.

These attacks are against the OEM's complete system, including both hardware and software. These systems operate in a wide variety of contexts. In addition, attackers have a broad range of motivations and skill sets. It is beyond the scope of this document to provide an exhaustive list of threats, mitigations, and contexts in which attacks might occur.

Consequently, this section focuses on trust architecture claims. This approach allows the OEM to assess how to best apply the trust architecture to mitigate attacks in a given context.

### 17.2 Characteristics and claims

1. Trust architecture is "opt in."
  - OEMs do not need to do anything to disable it. All trust features are enabled as the result of conscious decisions to set fuses, program registers, and perform code signing.
  - TrustZone is not entirely opt-in, as Arm cores come out of reset in execution level (EL) 3—TZ Secure World Supervisor level.

## Characteristics and claims

- Many SoC registers are only writeable when the CPU is executing in TZ Secure World.
  - Beyond basic configuration firmware which would be challenging to run outside of TZ Secure World, running security services in TZ Secure World is an OEM 'opt in' decision.
2. OEMs own all of the system's secrets.
    - OEMs are not dependent on NXP to provision devices, and NXP is not required to participate in the chain of trust for system manufacturing, deployment, or field servicing.
    - NXP is developing new software tools and services to support the chain of trust.
    - Trust 2.1 devices do include an NXP provisioned portion of a split key. Use of this split key is optional.
  3. Trust architecture allows OEMs to define system security policies & configurations, which cannot be changed or bypassed by attackers.
    - Always perform secure boot
    - Access control debug interfaces
    - Detect hardware security violations
    - Soft or Hard Fail reactions to security violations
  4. Trust architecture protects against unauthorized modifications to developer software and system configuration information (for example, device trees, certificates, and so on).
    - Protection consists of both prevention (secure boot) and after-the-fact detection mechanisms (for example, runtime integrity checking).
      - Secure boot detects unauthorized modifications and, when detected, prevents the unauthorized code from executing on the device.
      - Runtime integrity checking scans regions of memory for unauthorized modifications to the contents.
  5. Trust architecture allows OEMs to provision system permanent secrets, which are protected against extraction or exposure.
    - Permanent secrets continue across resets of the system, and are locked out in response to security violations.
    - Trust architecture 2.1 permanent secrets include:
      - Debug Response Value (DRV)
      - One-Time Programmable Master Key (OTPMK)
  6. Trust architecture allows OEMs to provision system persistent secrets which survive device resets under normal circumstances, but are capable of being cleared or rendered unusable.
    - Trust architecture 2.1 persistent secrets include:

- Battery backed Zeroizable Master Key (ZMK)
  - Any cryptographic blobs created with the OTPMK and stored to NVRAM prior to system reset
  - Cryptographic blobs can be used to make factory installed keys, certificates, and proprietary code into persistent secrets.
7. Trust architecture allows provisioning of system ephemeral secrets, which are protected against extraction or exposure.
- Ephemeral secrets are cleared by a security violation or system reset.
  - Trust architecture 2.1 ephemeral secrets include:
    - Job Descriptor Key Encryption Keys (JDKEKs)
    - Session keys negotiated during runtime, which are protected with the JDKEKs
    - Secrets owned by TZ Secure World trusted applications, and stored in private on-chip or off-chip memory are also considered ephemeral.
8. Trust architecture provides high levels of separation between independent software domains.
- The private resources of one domain must not be accessible by another domain.
  - These independent software partitions can be treated as independent security domains.
  - Arm TrustZone adds "Secure World," another level of separation not found in Power Architecture-based Trust Architecture devices.
9. Trust architecture 2.1 supports anti-rollback features
- Super Root Key revocation
  - Monotonic Counter

### 17.3 Non-claims

It is also important for OEMs to understand the types of attacks that NXP does not claim to prevent or strongly mitigate, so that mitigation can be taken at the system level if necessary.

1. Stronger than the underlying crypto algorithms.
  - Trust architecture's foundation rests on the strength of SHA-256, AES-256, and RSA digital signatures. If these algorithms are found to be fundamentally broken, most trust architecture claims are broken as well.
2. Preventing advanced physical attacks
  - As noted earlier, NXP recognizes that the contents of the Security Fuse Processor can be read by careful de-processing of the device. This attack destroys the QorIQ Layerscape device, however cryptographic blobs protected with the OTPMK could be recovered from system memory and decrypted.

- Memories and registers locked out or zeroized upon detection of security violations may be subject to memory remanence attacks.
- The SEC's PKHA implements timing equalization, and the AESA implements differential power analysis resistance features; however, the effectiveness of these features has not been independently evaluated. Other side channel attack methods, against the SEC and CPUs, may be possible given sufficient motivation.
- Operating the device outside of specified voltage, temperature, or frequency ranges can lead to unexpected failure modes. NXP has attempted to make all failure modes "fail safe," but the full range of possible glitches has not been evaluated.
- Trust architecture incorporates tamper detection inputs, which OEMs can use to alert the device to physical attacks. Without using the Zeroizable Master Key, the highest consequence of a security violation is that the OTPMK may be locked out. Assume that attackers are capable of bypassing external tamper detection logic and rebooting the system with no hardware security violations detected.

### 3. Providing absolute isolation between independent software domains/security domains

- Trust architecture's support for strong partitioning is based on access control mechanisms. If a resource is private to domain 1, domain 2 must not be able to access that resource, directly or indirectly. This is different from absolute isolation, in which domain 2 is unable to interfere with the operation of domain 1. From internal and external bus bandwidth consumption to unfiltered buffer releases, NXP knows of many scenarios in which domain 2 can interfere with domain 1.

### 4. Operating as a single-edged sword

- Trust architecture is designed to constrain the conditions under which the system can operate. Attackers can cause security violations for the purpose of shutting down the system. Trust architecture offers configuration options allowing OEMs to make trade-offs in security violation sensitivity and consequences, but it is beyond the scope of a trust architecture device to "know" when trust is being exploited to disrupt the system. Attackers in a position to trigger security violations could be capable of causing other types of system failures.

## 17.4 Related resources

QorIQ Trust Architecture is not a mandatory feature; it is opt in. To minimize the information available to potential attackers, access to detailed Trust Architecture reference material is restricted to vetted customers with active programs using Trust Architecture. Contact your NXP Sales or Field Application Engineer to initiate the admission process to the NXP QorIQ Trust Architecture User's Group site.

The following table shows related resources that may be helpful.

**Table 17-1. Related resources**

Resource	Purpose
<i>An Introduction to the QorIQ Platform's Trust Architecture (White Paper)</i>	Discusses the objectives of the trust architecture, and how it works. This white paper also includes logistical considerations.
<i>QorIQ Trust Architecture 2.1 User Guide</i>	Primary technical reference for Trust Architecture hardware blocks, including register level detail, as well as procedural steps to enable secure boot and security violation detection.
<i>User Enablement for Secure Boot</i>	<p>These documents are periodically updated as new features are added to the Code Signing Tool and reference chain of trust. They can be found in the QorIQ device SDK, and at <a href="http://www.nxp.com/infocenter">www.nxp.com/infocenter</a>: Software and Tools Information Center &gt; QorIQ SDK Documentation &gt; Secure Boot.</p> <p><b>NOTE:</b> Some of these documents are for PBL-based products, some are for non-PBL-based products. This chip is a PBL-based product.</p>

---

## Related resources

# Chapter 18

## DDR Memory Controller

### 18.1 DDR Introduction

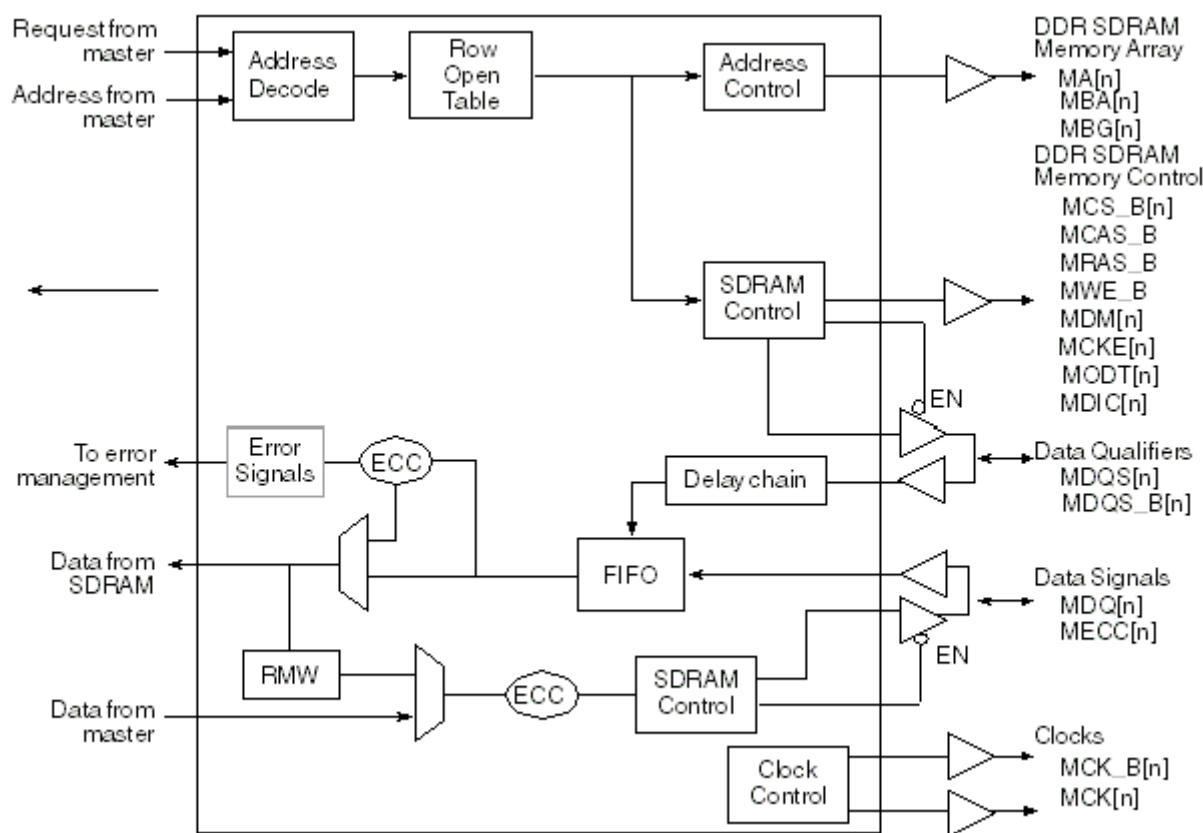
The DDR SDRAM controller controls processor and I/O interactions with system memory. The memory system supports a wide range of memory devices. Built-in error checking and correction (ECC) ensures very low bit-error rates for reliable high-frequency operation. Dynamic power management and auto-precharge modes simplify memory system design. A large set of special features, including ECC error injection, support rapid system debug.

#### NOTE

In this chapter, the word 'bank' refers to a physical bank specified by a chip select; 'logical bank' refers to one of the four or eight sub-banks in each SDRAM chip. A sub-bank is specified by the 2 or 3 bits on the bank address (MBA) pins during a memory access.

Also, since DDR3 and DDR3L behave identically except for voltage, all statements in this chapter that apply to DDR3 apply equally to DDR3L. Accordingly, DDR3 and DDR3L are collectively referred as 'DDR3 memory types' or 'DDR3 mode'.

This figure is a high-level block diagram of the DDR memory controller with its associated interfaces.



**Figure 18-1. DDR Memory Controller Simplified Block Diagram**

## 18.2 DDR Features

The DDR memory controller includes these distinctive features:

- Support for DDR standards:
  - DDR3L
  - DDR4

### NOTE

The DDR controller is designed to support DDR4 SDRAMs that are compliant with JESD79-4 from September 2012.

- Data bus widths supported:
  - 32-/36-bit
  - 16-/20-bit
- Programmable settings for meeting all SDRAM timing parameters
- The following SDRAM configurations are supported:

- As many as four physical banks (chip selects), each bank independently addressable
- 64-Mbit to 8-Gbit devices depending on internal device configuration
- 64-Mbit to 16-Gbit devices depending on internal device configuration
- Supports x8/x16 data ports
- Unbuffered DIMMs
- Support for data mask signals and read-modify-write for sub-double-word writes.  
Note that a read-modify-write sequence is only necessary when ECC is enabled.
- Support for double-bit error detection and single-bit error correction ECC (8-bit check word across 64-bit data)
- Support for address parity for DDR4 discrete memories
- Open page management (dedicated entry for each logical bank)
- Automatic DRAM initialization sequence or software-controlled initialization sequence
- Automatic DRAM data initialization
- Interrupt driven rapid clear of memory
- Write leveling supported
- Support for up to eight posted refreshes
- Memory controller clock frequency of two times the SDRAM clock with support for sleep power management
- Support for error injection

### 18.2.1 DDR Modes of Operation

The DDR memory controller supports the following modes:

- Dynamic power management mode. The DDR memory controller can reduce power consumption by negating the SDRAM CKE signal when no transactions are pending to the SDRAM.
- Auto-precharge mode. Clearing DDR\_SDRAM\_INTERVAL[BSTOPRE] causes the memory controller to issue an auto-precharge command with every read or write transaction. Auto-precharge mode can be enabled for separate chip selects by setting CS<sub>n</sub>\_CONFIG[AP<sub>n</sub>\_EN].

## 18.3 DDR External Signal Descriptions

This section describes external signals of the DDR memory controller. It describes each signal's behavior when the signal is asserted or negated and when the signal is an input or an output.

### 18.3.1 DDR Signals Overview

Memory controller signals are grouped as follows:

- Memory interface signals
- Clock signals

This table shows how DDR memory controller external signals are grouped. The device data sheet has a pinout diagram showing pin numbers. It also lists all electrical and mechanical specifications.

#### NOTE

When ECC DRAM is present ECC should be enabled.

**Table 18-1. DDR Memory Interface Signal Summary**

Name	Function/Description	Reset	Pins	I/O
MAPAR_ERR_B	Address parity error	One	1	I
MAPAR_OUT	Address parity out	Zero	1	O
MDQ[0:31]	Data bus	All zeros	32	I/O
MDQS[0:3], MDQS8	Data strobes	All zeros	5	I/O
MDQS[0:3]_B, MDQS8_B	Complement data strobes	All ones	5	I/O
MECC[0:3]	Error checking and correcting	All zeros	4	I/O
MCAS_B/MA15	Column address strobe	One	1	O
MA15/MACT_B, MA14/MBG1, MA[13:0]	Address bus	All zeros	16	O
MBA2/MBG0, MBA[1:0]	Logical bank address	All zeros	3	O
MCS[0:3]_B	Chip selects	All ones	4	O
MWE_B/MA14	Write enable	One	1	O
MRAS_B/MA16	Row address strobe	One	1	O
MDM[0:3]/MDBI[0:3], MDM8/MDBI8	Data mask	All zeros	5	O
MCK[0:1]	DRAM clock outputs NUM_MCK	All zeros	2	O
MCK[0:1]_B	DRAM clock outputs (complement)	All zeros	2	O
MCKE[0:1]	DRAM clock enable	All zeros	2	O
MODT[0:1]	DRAM on-die termination	All zeros	2	O
MDIC[0:1]	Driver impedance calibration	See chip datasheet for more information	2	I/O

This table shows the memory address signal mappings for DDR3 memory types.

**Table 18-2. Memory Address Signal Mappings for DDR3 Memory Types**

Controller Signal Name (Outputs)		DRAM/DIMM Signal Name (Inputs)
msb	MA15	A15
	MA14	A14
	MA13	A13
	MA12	A12
	MA11	A11
	MA10	A10 (AP for DDR)
	MA9	A9
	MA8	A8
	MA7	A7
	MA6	A6
	MA5	A5
	MA4	A4
	MA3	A3
	MA2	A2
	MA1	A1
lsb	MA0	A0
msb	MBA2	BA2
	MBA1	BA1
lsb	MBA0	BA0
	MRAS_B	RAS_B
	MCAS_B	CAS_B
	MWE_B	WE_B
msb	MCS0_B	CS0_B
	MCS1_B	CS1_B
	MCS2_B	CS2_B
lsb	MCS3_B	CS3_B

This table shows the memory address signal mappings for DDR4.

**Table 18-3. Memory Address Signal Mappings for DDR4**

Controller Signal Name (Outputs)		DRAM/DIMM Signal Name (Inputs)
	MA15/MACT_B	ACT_n
	MA14/MBG1	BG1
msb	MA13	A13
	MA12	A12
	MA11	A11
	MA10	A10 (AP for DDR)

*Table continues on the next page...*

**Table 18-3. Memory Address Signal Mappings for DDR4 (continued)**

Controller Signal Name (Outputs)	DRAM/DIMM Signal Name (Inputs)	
	MA9	A9
	MA8	A8
	MA7	A7
	MA6	A6
	MA5	A5
	MA4	A4
	MA3	A3
	MA2	A2
	MA1	A1
lsb	MA0	A0
msb	MBA2	BG0
	MBA1	BA1
lsb	MBA0	BA0
	MRAS_B/MA16	RAS_n/A16
	MCAS_B/MA15	CAS_n/A15
	MWE_B/MA14	WE_n/A14
msb	MCS0_B	CS0_n
	MCS1_B	CS1_n
	MCS2_B	CS2_n
lsb	MCS3_B	CS3_n
	MODT1	ODT1
	MDM[0:3]/MDBI[0:3], MDM8/MDBI8	DM_n[0:3]/DBI_n[0:3], DM_n[8]/DBI_n[8]
	MAPAR_ERR_B	ALERT_n
	MAPAR_OUT	PAR

## 18.3.2 DDR Detailed Signal Descriptions

The following sections describe the DDR SDRAM controller input and output signals, the meaning of their different states, and relative timing information for assertion and negation.

### 18.3.2.1 Memory Interface Signals

This table describes the DDR controller memory interface signals.

**Table 18-4. Memory Interface Signals-Detailed Signal Descriptions**

Signal	I/O	Description	
MDQ[0:31]	I/O	Data bus. Both input and output signals on the DDR memory controller.	
		As outputs for the bidirectional data bus, these signals operate as described below.	
		<b>State Meaning</b>	Asserted/Negated - Represent the value of data being driven by the DDR memory controller.
		<b>Timing</b>	Assertion/Negation - Driven with valid data during writes to memory. High impedance-No READ or WRITE command is in progress; data is not being driven by the memory controller or the DRAM.
	I	As inputs for the bidirectional data bus, these signals operate as described below.	
		<b>State Meaning</b>	Asserted/Negated - Represents the state of data being driven by the external DDR SDRAMs.
		<b>Timing</b>	Assertion/Negation - The DDR SDRAM drives data during a READ transaction. High impedance-No READ or WRITE command in progress; data is not being driven by the memory controller or the DRAM.
MDQS[0:3], MDQS8 MDQS[0:3]_B, MDQS8_B	I/O	Data strobes. Inputs with read data, outputs with write data. The data strobes must be differential.	
		As outputs, the data strobes are driven by the DDR memory controller during a write transaction.	
		<b>State Meaning</b>	Asserted/Negated - Driven high when positive capture data is transmitted and driven low when negative capture data is transmitted. Centered in the data "eye" for writes; coincident with the data eye for reads. Treated as a clock. Data is valid when signals toggle. See <a href="#">DDR SDRAM Interface Operation</a> for byte lane assignments.
		<b>Timing</b>	Assertion/Negation - If a WRITE command is registered at clock edge $n$ , data strobes at the DRAM assert centered in the data eye on clock edge $n + 1$ . See the JEDEC DDR SDRAM specification for more information.
	I	As inputs, the data strobes are driven by the external DDR SDRAMs during a read transaction. The data strobes are used by the memory controller to synchronize data latching.	
		<b>State Meaning</b>	Asserted/Negated - Driven high when positive capture data is received and driven low when negative capture data is received. Centered in the data eye for writes; coincident with the data eye for reads. Treated as a clock. Data is valid when signals toggle. See <a href="#">DDR SDRAM Interface Operation</a> for byte lane assignments.
		<b>Timing</b>	Assertion/Negation - If a READ command is registered at clock edge $n$ , and the latency is programmed in TIMING_CFG_1[CASLAT] to be $m$ clocks, data strobes at the DRAM assert coincident with the data on clock edge $n + m$ . See the JEDEC DDR SDRAM specification for more information.
MECC[0:3]	I/O	Error checking and correcting codes. Input and output signals for the DDR controller's bidirectional ECC bus.	
		As normal mode outputs the ECC signals represent the state of ECC driven by the DDR controller on writes.	
		<b>State Meaning</b>	Asserted/Negated - Represents the state of ECC being driven by the DDR controller on writes.
		<b>Timing</b>	Assertion/Negation - Same timing as MDQ High impedance - Same timing as MDQ
	I	As inputs, the ECC signals represent the state of ECC driven by the SDRAM devices on reads.	
		<b>State Meaning</b>	Asserted/Negated - Represents the state of ECC being driven by the DDR SDRAMs on reads.
		<b>Timing</b>	Assertion/Negation - Same timing as MDQ High impedance-Same timing as MDQ

*Table continues on the next page...*

**Table 18-4. Memory Interface Signals-Detailed Signal Descriptions (continued)**

Signal	I/O	Description				
MA15/MACT_B MA14/MBG1 MA[13:0]	O	<p>Address bus. Memory controller outputs for the address to the DRAM. MAn carry the address bits for the DDR memory interface comprising the row and column address bits. MA0 is the lsb of the address output from the memory controller.</p> <p>In DDR4 mode: Some of these signals are used for other purposes:</p> <ul style="list-style-type: none"> <li>The activate signal (ACT_n) is driven on MA15/MACT_B. It is never an address signal.</li> <li>The BG1 signal is driven on MA14/MBG1. It is never an address signal.</li> </ul> <p>In DDR3 mode, MA15/MACT_B and MA14/MBG1 are always address signals</p> <p>In DDR4 mode: During activate command assertion, address signal 15 is driven on MCAS_B/MA15 and address signal 14 is driven on MWE_B/MA14.</p>				
		<table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted/Negated - Represents the address driven by the DDR memory controller. Contains different portions of the address depending on the memory size and the DRAM command being issued by the memory controller. See <a href="#">DDR SDRAM Address Multiplexing</a> for a complete description of the mapping of these signals.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - The address is always driven when the memory controller is enabled. It is valid when a transaction is driven to DRAM (when MCS_Bn is active).  High impedance - When the memory controller is disabled</td></tr> </table>	<b>State Meaning</b>	Asserted/Negated - Represents the address driven by the DDR memory controller. Contains different portions of the address depending on the memory size and the DRAM command being issued by the memory controller. See <a href="#">DDR SDRAM Address Multiplexing</a> for a complete description of the mapping of these signals.	<b>Timing</b>	Assertion/Negation - The address is always driven when the memory controller is enabled. It is valid when a transaction is driven to DRAM (when MCS_Bn is active).  High impedance - When the memory controller is disabled
<b>State Meaning</b>	Asserted/Negated - Represents the address driven by the DDR memory controller. Contains different portions of the address depending on the memory size and the DRAM command being issued by the memory controller. See <a href="#">DDR SDRAM Address Multiplexing</a> for a complete description of the mapping of these signals.					
<b>Timing</b>	Assertion/Negation - The address is always driven when the memory controller is enabled. It is valid when a transaction is driven to DRAM (when MCS_Bn is active).  High impedance - When the memory controller is disabled					
MBA2/MBG0, MBA[1:0]	O	<p>Logical bank address. Outputs that drive the logical (or internal) bank address pins of the SDRAM. Each SDRAM supports four or eight addressable logical sub-banks. Bit zero of the memory controller's output bank address must be connected to bit zero of the SDRAM's input bank address. MBA0, the least-significant bit of the bank address signals, is asserted during the mode register set command to specify the extended mode register.</p> <p>In DDR4 mode, MBA2 should be connected to BG0 on the DRAM.</p>				
		<table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted/Negated - Selects the DDR SDRAM logical (or internal) bank to be activated during the row address phase and selects the SDRAM internal bank for the read or write operation during the column address phase of the memory access. <a href="#">DDR SDRAM Address Multiplexing</a> describes the mapping of these signals in all cases.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - Same timing as MAn  High impedance - Same timing as MAn</td></tr> </table>	<b>State Meaning</b>	Asserted/Negated - Selects the DDR SDRAM logical (or internal) bank to be activated during the row address phase and selects the SDRAM internal bank for the read or write operation during the column address phase of the memory access. <a href="#">DDR SDRAM Address Multiplexing</a> describes the mapping of these signals in all cases.	<b>Timing</b>	Assertion/Negation - Same timing as MAn  High impedance - Same timing as MAn
<b>State Meaning</b>	Asserted/Negated - Selects the DDR SDRAM logical (or internal) bank to be activated during the row address phase and selects the SDRAM internal bank for the read or write operation during the column address phase of the memory access. <a href="#">DDR SDRAM Address Multiplexing</a> describes the mapping of these signals in all cases.					
<b>Timing</b>	Assertion/Negation - Same timing as MAn  High impedance - Same timing as MAn					
MCAS_B/MA15	O	<p>Column address strobe. Active-low SDRAM address multiplexing signal. MCAS_B is asserted for read or write transactions and for mode register set, refresh, self-refrsh and precharge commands in DDR3 mode .</p> <p>Note that in DDR3 mode this signal is only used as MCASB: MA15 has a dedicated signal in DDR3 mode.</p> <p>In DDR4 mode: This signal is multiplexed with MA15. During activate command assertion, MA15 is driven on this pin; see MAn for details of state meaning and timing. At all other times, the pin is driven as MCAS_B, with state meaning and timing as shown below.</p>				
		<table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted - Indicates that a valid SDRAM column address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MCAS_B for various other SDRAM commands.  Negated - The column address is not guaranteed to be valid.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a>, <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>, <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a>, and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MCAS_B is always driven unless the memory controller is disabled.</td></tr> </table>	<b>State Meaning</b>	Asserted - Indicates that a valid SDRAM column address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MCAS_B for various other SDRAM commands.  Negated - The column address is not guaranteed to be valid.	<b>Timing</b>	Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a> , <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a> , <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a> , and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MCAS_B is always driven unless the memory controller is disabled.
<b>State Meaning</b>	Asserted - Indicates that a valid SDRAM column address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MCAS_B for various other SDRAM commands.  Negated - The column address is not guaranteed to be valid.					
<b>Timing</b>	Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a> , <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a> , <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a> , and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MCAS_B is always driven unless the memory controller is disabled.					

*Table continues on the next page...*

**Table 18-4. Memory Interface Signals-Detailed Signal Descriptions (continued)**

Signal	I/O	Description				
MRAS_B/MA16	O	<p>Row address strobe. Active-low SDRAM address multiplexing signal. Asserted for activate, refresh, self-refresh and precharge commands in DDR3 mode.</p> <p>Note that in DDR3 mode this signal is only used as MRAS_B.</p> <p>In DDR4 mode: This signal is multiplexed with MA16. During activate command assertion, MA16 is driven on this pin; see MAn for details of state meaning and timing. At all other times, the pin is driven as MRAS_B, with state meaning and timing as shown below.</p> <table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted - Indicates that a valid SDRAM row address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MRAS_B for various other SDRAM commands.  Negated - The row address is not guaranteed to be valid.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a>, <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>, <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a>, and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MRAS_B is always driven unless the memory controller is disabled.</td></tr> </table>	<b>State Meaning</b>	Asserted - Indicates that a valid SDRAM row address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MRAS_B for various other SDRAM commands.  Negated - The row address is not guaranteed to be valid.	<b>Timing</b>	Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a> , <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a> , <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a> , and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MRAS_B is always driven unless the memory controller is disabled.
<b>State Meaning</b>	Asserted - Indicates that a valid SDRAM row address is on the address bus for read and write transactions. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MRAS_B for various other SDRAM commands.  Negated - The row address is not guaranteed to be valid.					
<b>Timing</b>	Assertion/Negation - Assertion and negation timing is directed by the values described in <a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a> , <a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a> , <a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a> , and <a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>  High impedance - MRAS_B is always driven unless the memory controller is disabled.					
MCS[0:3]_B	O	<p>Chip selects. Four chip selects supported by the memory controller.</p> <table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted - Selects a physical SDRAM bank to perform a memory operation as described in <a href="#">Chip select a memory bounds (CS0_BNDS - CS3_BNDS)</a>, and <a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>. The DDR controller asserts one of the MCS_B[0:3]signals to begin a memory cycle.  Negated - Indicates no SDRAM action during the current cycle.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - Asserted to signal any new transaction to the SDRAM. The transaction must adhere to the timing constraints set in TIMING_CFG_0-TIMING_CFG_3.  High impedance - Always driven unless the memory controller is disabled.</td></tr> </table>	<b>State Meaning</b>	Asserted - Selects a physical SDRAM bank to perform a memory operation as described in <a href="#">Chip select a memory bounds (CS0_BNDS - CS3_BNDS)</a> , and <a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a> . The DDR controller asserts one of the MCS_B[0:3]signals to begin a memory cycle.  Negated - Indicates no SDRAM action during the current cycle.	<b>Timing</b>	Assertion/Negation - Asserted to signal any new transaction to the SDRAM. The transaction must adhere to the timing constraints set in TIMING_CFG_0-TIMING_CFG_3.  High impedance - Always driven unless the memory controller is disabled.
<b>State Meaning</b>	Asserted - Selects a physical SDRAM bank to perform a memory operation as described in <a href="#">Chip select a memory bounds (CS0_BNDS - CS3_BNDS)</a> , and <a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a> . The DDR controller asserts one of the MCS_B[0:3]signals to begin a memory cycle.  Negated - Indicates no SDRAM action during the current cycle.					
<b>Timing</b>	Assertion/Negation - Asserted to signal any new transaction to the SDRAM. The transaction must adhere to the timing constraints set in TIMING_CFG_0-TIMING_CFG_3.  High impedance - Always driven unless the memory controller is disabled.					
MWE_B/MA14	O	<p>Write enable. Asserted when a write transaction is issued to the SDRAM. This is also used for mode register set commands and precharge commands in DDR3 mode .</p> <p>Note that in DDR3 mode this signal is only used as MWE_B: MA14 has a dedicated signal in DDR3 mode.</p> <p>In DDR4 mode: This signal is multiplexed with MA14. During activate command assertion, MA14 is driven on this pin; see MAn for details of state meaning and timing. At all other times, the pin is driven as MWE_B, with state meaning and timing as shown below.</p> <table border="1"> <tr> <td><b>State Meaning</b></td><td>Asserted - Indicates a memory write operation. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MWE_B for various other SDRAM commands.  Negated - Indicates a memory read operation.</td></tr> <tr> <td><b>Timing</b></td><td>Assertion/Negation - Similar timing as MRAS_B and MCAS_B. Used for write commands.  High impedance - MWE_B is always driven unless the memory controller is disabled.</td></tr> </table>	<b>State Meaning</b>	Asserted - Indicates a memory write operation. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MWE_B for various other SDRAM commands.  Negated - Indicates a memory read operation.	<b>Timing</b>	Assertion/Negation - Similar timing as MRAS_B and MCAS_B. Used for write commands.  High impedance - MWE_B is always driven unless the memory controller is disabled.
<b>State Meaning</b>	Asserted - Indicates a memory write operation. Refer to the JEDEC DDR SDRAM specifications for more information on the states required on MWE_B for various other SDRAM commands.  Negated - Indicates a memory read operation.					
<b>Timing</b>	Assertion/Negation - Similar timing as MRAS_B and MCAS_B. Used for write commands.  High impedance - MWE_B is always driven unless the memory controller is disabled.					
MDM[0:3]/MDBI[0:3], MDM8/MDBI8	I/O	<p>These signals are multiplexed in DDR4. They can be either DDR SDRAM data output masks or data bus inversion signals. (See <a href="#">DDR SDRAM control configuration 3 (DDR_SDRAM_CFG_3)</a> for more information.) As data masks MDMn, these signals mask unwanted bytes of data transferred during a write. They are needed to support sub-burst-size transactions (such as single-byte writes) on SDRAM where all I/O occurs in multi-byte bursts. MDM0 corresponds to the most significant byte (MSB) and MDM3 corresponds to the LSB, while MDM8 corresponds to the ECC byte. <a href="#">DDR SDRAM Interface Operation</a> shows byte lane encodings.</p> <p>Note that in DDR3 mode these signals are only used as MDMn: data bus inversion is a DDR4-only feature.</p>				

*Table continues on the next page...*

**Table 18-4. Memory Interface Signals-Detailed Signal Descriptions (continued)**

Signal	I/O	Description
	O	As DDR SDRAM data bus inversion signals DBIn_n, these signals indicate that all the data bits of the byte lane are inverted on the data bus during a read or write transaction. When used for this purpose, the signals are I/Os.
		As outputs, these signals operate as described below.
		<p><b>State Meaning</b> Asserted-As masks, prevent writing to DDR SDRAM. Asserted when data is written to DRAM if the corresponding byte(s) should be masked for the write. Note that the MDMn signals are active-high for the DDR controller. MDMn is part of the DDR command encoding. As DBIs, indicate bus signal inversion per byte lane; asserted by the controller during a write cycle. Note that the DBIn_n signals are active-low for both the DRAM and DDR controller.</p> <p>Negated-As masks, allow the corresponding byte to be read from or written to the SDRAM. As DBIs, signal DRAMs or the DDR controller to invert all DQx in that byte lane before storing data.</p>
		<p><b>Timing</b> Assertion/Negation-Same timing as MDQx as outputs.</p> <p>High impedance-Always driven unless the memory controller is disabled.</p>
	I	As inputs, these signals operate as described below.
		<p><b>State Meaning</b> Asserted-As DBIs, indicate bus signal inversion per byte lane; asserted by the DRAMs during a read cycle. Note that the DBIn_n signals are active-low for both the DRAM and DDR controller.</p> <p>Negated-As DBIs, signal DRAMs or the DDR controller to invert all DQx in that byte lane before storing data.</p>
		<p><b>Timing</b> Assertion/Negation-Same timing as MDQx as inputs.</p> <p>High impedance-Always driven unless the memory controller is disabled.</p>
MODT[0:1]	O	On-Die termination. Memory controller outputs for the ODT to the DRAM. MODT represents the on-die termination for the associated data, data masks, ECC, and data strobes. The MODT signal should be connected to the same rank of memory as the corresponding MCK/MCK_B pair.
		<p><b>State Meaning</b> Asserted/Negated - Represents the ODT driven by the DDR memory controller.</p>
		<p><b>Timing</b> Assertion/Negation - Driven in accordance with JEDEC DRAM specifications for on-die termination timings. It is configured through the CSn_CONFIG[ODT_RD_CFG] and CSn_CONFIG[ODT_WR_CFG] fields.</p> <p>High impedance - Always driven.</p>
MDIC[0:1]	I/O	Driver impedance calibration. The proper value and connection of the MDIC resistor are specified in the corresponding device data sheet document. See <a href="#">DDR Control Driver Register 1 (DDRCDR_1)</a> for more information on these signals.
		<p><b>State Meaning</b> Used for automatic calibration of the DDR IOs</p>
		<p><b>Timing</b> Driven for four DRAM cycles at a time while the DDR controller is executing the automatic driver compensation</p>
MAPAR_ERR_B	I	Address parity error. Reflects whether an address parity error has been detected by the DRAM. This signal is active low. MAPAR_ERR_B is used in DDR4 mode.
		<p><b>State Meaning</b> Asserted - An error has been detected. Negated - An error has not been detected.</p>
		<p><b>Timing</b> Assertion/Negation- In DDR4 mode: Driven by DRAM ALERT_n pin a period of time after parity error has been detected by the DRAM.</p>

Table continues on the next page...

**Table 18-4. Memory Interface Signals-Detailed Signal Descriptions (continued)**

Signal	I/O	Description
MAPAR_OUT	O	Address parity out. Driven by the memory controller as the parity bit calculated across the address and command bits. Even parity is used, and parity is not calculated for the MCKE[0:3], MODT[0:3], or MCS_B[0:3] signals. MAPAR_OUT can be used with any DDR4 DRAM.
		<b>State Meaning</b> Asserted-The parity bit is high. Negated-The parity bit is low.
		<b>Timing</b> Assertion/Negation-In DDR3 mode: Will be issued one DRAM cycle after the chip select for each command. In DDR4 mode: Will be issued at the same cycle as the chip select.

### 18.3.2.2 Clock Interface Signals

This table contains the detailed descriptions of the clock signals of the DDR controller.

**Table 18-5. Clock Signals - Detailed Signal Descriptions**

Signal	I/O	Description
MCK[0:1], MCK_B[0:1]	O	DRAM clock output and complement.
		<b>State Meaning</b> Asserted/Negated - The JEDEC DDR SDRAM specifications require true and complement clocks. A clock edge is seen by the SDRAM when the true and complement cross.
		<b>Timing</b> Assertion/Negation - Timing is controlled by the DDR_CLK_CNTL register at offset 0x130.
MCKE[0:1]	O	Clock enable. Output signal used as the clock enable to the SDRAM. MCKE can be negated to stop clocking the DDR SDRAM. The MCKE signal should be connected to the same rank of memory as the corresponding MCK/MCK_B pair.
		<b>State Meaning</b> Asserted - Clocking to the SDRAM is enabled. Negated - Clocking to the SDRAM is disabled and the SDRAM should ignore signal transitions on MCK or MCK_B. MCK/MCK_B are don't cares while MCKE is negated.
		<b>Timing</b> Assertion/Negation - Asserted when DDR_SDRAM_CFG[MEM_EN] is set. Can be negated when entering dynamic power management or self refresh. Will be asserted again when exiting dynamic power management or self refresh. High impedance-Always driven.

## 18.4 DDR register descriptions

This table shows the register memory map for the DDR memory controller.

### 18.4.1 DDR memory map

DDR base address: 108\_0000h

## DDR register descriptions

Offset	Register	Width (In bits)	Access	Reset value
0h	Chip select 0 memory bounds (CS0_BNDS)	32	RW	0000_0000h
8h	Chip select 1 memory bounds (CS1_BNDS)	32	RW	0000_0000h
10h	Chip select 2 memory bounds (CS2_BNDS)	32	RW	0000_0000h
18h	Chip select 3 memory bounds (CS3_BNDS)	32	RW	0000_0000h
80h - 8Ch	Chip select a configuration (CS0_CONFIG - CS3_CONFIG)	32	RW	0000_0000h
100h	DDR SDRAM timing configuration 3 (TIMING_CFG_3)	32	RW	0000_0000h
104h	DDR SDRAM timing configuration 0 (TIMING_CFG_0)	32	RW	0011_0005h
108h	DDR SDRAM timing configuration 1 (TIMING_CFG_1)	32	RW	0000_0000h
10Ch	DDR SDRAM timing configuration 2 (TIMING_CFG_2)	32	RW	0000_0000h
110h	DDR SDRAM control configuration (DDR_SDRAM_CFG)	32	RW	0700_0000h
114h	DDR SDRAM control configuration 2 (DDR_SDRAM_CFG_2)	32	RW	0000_0000h
118h	DDR SDRAM mode configuration (DDR_SDRAM_MODE)	32	RW	0000_0000h
11Ch	DDR SDRAM mode configuration 2 (DDR_SDRAM_MODE_2)	32	RW	0000_0000h
120h	DDR SDRAM mode control (DDR_SDRAM_MD_CNTL)	32	RW	0000_0000h
124h	DDR SDRAM interval configuration (DDR_SDRAM_INTERVAL)	32	RW	0000_0000h
128h	DDR SDRAM data initialization (DDR_DATA_INIT)	32	RW	0000_0000h
130h	DDR SDRAM clock control (DDR_SDRAM_CLK_CNTL)	32	RW	0200_0000h
148h	DDR training initialization address (DDR_INIT_ADDR)	32	RW	0000_0000h
14Ch	DDR training initialization extended address (DDR_INIT_EXT_ADDRESS)	32	RW	0000_0000h
160h	DDR SDRAM timing configuration 4 (TIMING_CFG_4)	32	RW	0000_0000h
164h	DDR SDRAM timing configuration 5 (TIMING_CFG_5)	32	RW	0000_0000h
168h	DDR SDRAM timing configuration 6 (TIMING_CFG_6)	32	RW	0000_0000h
16Ch	DDR SDRAM timing configuration 7 (TIMING_CFG_7)	32	RW	0000_0000h
170h	DDR ZQ calibration control (DDR_ZQ_CNTL)	32	RW	0000_0000h
174h	DDR write leveling control (DDR_WRLVL_CNTL)	32	RW	0000_0000h
17Ch	DDR Self Refresh Counter (DDR_SR_CNTR)	32	RW	0000_0000h
180h	DDR Register Control Words 1 (DDR_SDRAM_RCW_1)	32	RW	0000_0000h
184h	DDR Register Control Words 2 (DDR_SDRAM_RCW_2)	32	RW	0000_0000h
190h	DDR write leveling control 2 (DDR_WRLVL_CNTL_2)	32	RW	0000_0000h
194h	DDR write leveling control 3 (DDR_WRLVL_CNTL_3)	32	RW	0000_0000h
1A0h	DDR Register Control Words 3 (DDR_SDRAM_RCW_3)	32	RW	0000_0000h
1A4h	DDR Register Control Words 4 (DDR_SDRAM_RCW_4)	32	RW	0000_0000h
1A8h	DDR Register Control Words 5 (DDR_SDRAM_RCW_5)	32	RW	0000_0000h
1ACh	DDR Register Control Words 6 (DDR_SDRAM_RCW_6)	32	RW	0000_0000h
200h	DDR SDRAM mode configuration 3 (DDR_SDRAM_MODE_3)	32	RW	0000_0000h
204h	DDR SDRAM mode configuration 4 (DDR_SDRAM_MODE_4)	32	RW	0000_0000h
208h	DDR SDRAM mode configuration 5 (DDR_SDRAM_MODE_5)	32	RW	0000_0000h
20Ch	DDR SDRAM mode configuration 6 (DDR_SDRAM_MODE_6)	32	RW	0000_0000h
210h	DDR SDRAM mode configuration 7 (DDR_SDRAM_MODE_7)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
214h	DDR SDRAM mode configuration 8 (DDR_SDRAM_MODE_8)	32	RW	0000_0000h
220h	DDR SDRAM mode configuration 9 (DDR_SDRAM_MODE_9)	32	RW	0000_0000h
224h	DDR SDRAM mode configuration 10 (DDR_SDRAM_MODE_10)	32	RW	0000_0000h
228h	DDR SDRAM mode configuration 11 (DDR_SDRAM_MODE_11)	32	RW	0000_0000h
22Ch	DDR SDRAM mode configuration 12 (DDR_SDRAM_MODE_12)	32	RW	0000_0000h
230h	DDR SDRAM mode configuration 13 (DDR_SDRAM_MODE_13)	32	RW	0000_0000h
234h	DDR SDRAM mode configuration 14 (DDR_SDRAM_MODE_14)	32	RW	0000_0000h
238h	DDR SDRAM mode configuration 15 (DDR_SDRAM_MODE_15)	32	RW	0000_0000h
23Ch	DDR SDRAM mode configuration 16 (DDR_SDRAM_MODE_16)	32	RW	0000_0000h
250h	DDR SDRAM timing configuration 8 (TIMING_CFG_8)	32	RW	0000_0000h
260h	DDR SDRAM control configuration 3 (DDR_SDRAM_CFG_3)	32	RW	0000_0000h
400h	DQ mapping register 0 (DDR_DQ_MAP0)	32	RW	0000_0000h
404h	DQ mapping register 1 (DDR_DQ_MAP1)	32	RW	0000_0000h
408h	DQ mapping register 2 (DDR_DQ_MAP2)	32	RW	0000_0000h
40Ch	DQ mapping register 3 (DDR_DQ_MAP3)	32	RW	0000_0000h
B20h	DDR Debug Status Register 1 (DDRDSR_1)	32	RO	0000_8080h
B24h	DDR Debug Status Register 2 (DDRDSR_2)	32	RW	8000_0000h
B28h	DDR Control Driver Register 1 (DDRCDR_1)	32	RW	0000_0000h
B2Ch	DDR Control Driver Register 2 (DDRCDR_2)	32	RW	0000_0000h
BF8h	DDR IP block revision 1 (DDR_IP_REV1)	32	RO	0002_0501h
BFCCh	DDR IP block revision 2 (DDR_IP_REV2)	32	RO	0000_0000h
D00h	DDR Memory Test Control Register (DDR_MTCSR)	32	RW	0000_0000h
D20h - D44h	DDR Memory Test Pattern n Register (DDR_MTP0 - DDR_MTP9)	32	RW	0000_0000h
D60h	DDR Memory Test Start Extended Address (DDR_MT_ST_EXT_ADDR)	32	RW	0000_0000h
D64h	DDR Memory Test Start Address (DDR_MT_ST_ADDR)	32	RW	0000_0000h
D68h	DDR Memory Test End Extended Address (DDR_MT_END_EXT_ADDR)	32	RW	0000_0000h
D6Ch	DDR Memory Test End Address (DDR_MT_END_ADDR)	32	RW	0000_0000h
E00h	Memory data path error injection mask high (DATA_ERR_INJECT_HI)	32	RW	0000_0000h
E04h	Memory data path error injection mask low (DATA_ERR_INJECT_LO)	32	RW	0000_0000h
E08h	Memory data path error injection mask ECC (ECC_ERR_INJECT)	32	RW	0000_0000h
E20h	Memory data path read capture high (CAPTURE_DATA_HI)	32	RW	0000_0000h
E24h	Memory data path read capture low (CAPTURE_DATA_LO)	32	RW	0000_0000h
E28h	Memory data path read capture ECC (CAPTURE_ECC)	32	RW	0000_0000h
E40h	Memory error detect (ERR_DETECT)	32	W1C	0000_0000h
E44h	Memory error disable (ERR_DISABLE)	32	RW	0000_0000h
E48h	Memory error interrupt enable (ERR_INT_EN)	32	RW	0000_0000h
E4Ch	Memory error attributes capture (CAPTURE_ATTRIBUTES)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
E50h	Memory error address capture (CAPTURE_ADDRESS)	32	RW	0000_0000h
E54h	Memory error extended address capture (CAPTURE_EXT_ADDR_ESS)	32	RW	0000_0000h
E58h	Single-Bit ECC memory error management (ERR_SBE)	32	RW	0000_0000h

## 18.4.2 Chip select a memory bounds (CS0\_BNDS - CS3\_BNDS)

### 18.4.2.1 Offset

Register	Offset
CS0_BNDS	0h
CS1_BNDS	8h
CS2_BNDS	10h
CS3_BNDS	18h

### 18.4.2.2 Function

The chip select bounds registers (CS $n$  \_BNDS) define the starting and ending address of the memory space that corresponds to the individual chip selects. Note that the size specified in CS $n$  \_BNDS should equal the size of physical DRAM. Also, note that EAn must be greater than or equal to SAn.

If chip select interleaving is enabled, all fields in the lower interleaved chip select will be used, and the other chip selects' bounds registers will be unused. For example, if chip selects 0 and 1 are interleaved, all fields in CS0\_BNDS will be used, and all fields in CS1\_BNDS will be unused.

### 18.4.2.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									SA							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									EA							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.2.4 Fields

Field	Function
0-15 SA	Starting Address. Starting address for chip select (bank) $n$ . This value is compared against the 16 msbs of the 40-bit address.
16-31 EA	Ending Address. Ending address for chip select (bank) $n$ . This value is compared against the 16 msbs of the 40-bit address.

## 18.4.3 Chip select a configuration (CS0\_CONFIG - CS3\_CONFIG)

### 18.4.3.1 Offset

Register	Offset
CS0_CONFIG	80h
CS1_CONFIG	84h
CS2_CONFIG	88h
CS3_CONFIG	8Ch

### 18.4.3.2 Function

The chip select configuration ( $CS_n\_CONFIG$ ) registers enable the DDR chip selects and set the number of row and column bits used for each chip select. These registers should be loaded with the correct number of row and column bits for each SDRAM. Because  $CS_n\_CONFIG[ROW\_BITS\_CS_n, COL\_BITS\_CS_n]$  establish address multiplexing, the user should take great care to set these values correctly.

If chip select interleaving is enabled, then all fields in the lower interleaved chip select will be used, and the other registers' fields will be unused, with the exception of the ODT\_RD\_CFG and ODT\_WR\_CFG fields. For example, if chip selects 0 and 1 are interleaved, all fields in CS0\_CONFIG will be used, but only the ODT\_RD\_CFG and ODT\_WR\_CFG fields in CS1\_CONFIG will be used.

### 18.4.3.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CS_EN	Reserved							AP_EN	ODT_RD_CFG	Reserved			ODT_WR_CFG		
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	BA_BITS_CS	Reserved		ROW_BITS_CS			Reserved		BG_BITS_CS		Reserved		COL_BITS_CS			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.3.4 Fields

Field	Function
0 CS_EN	Chip Select Enable. Chip select nable 0b - Chip select nis not active 1b - Chip select nis active and assumes the state set in CSn_BNDS.
1-7	Reserved

Table continues on the next page...

Field	Function
—	
8 AP_EN	<p>Auto Precharge Enable.</p> <p>Chip select <i>n</i>auto-precharge enable</p> <p>0b - Chip select will only be auto-precharged if global auto-precharge mode is enabled (DDR_SDRAM_INTERVAL[BSTOPRE] = 0).</p> <p>1b - Chip select will always issue an auto-precharge for read and write transactions.</p>
9-11 ODT_RD_CFG	<p>On-Die Termination Read Config.</p> <p>ODT for reads configuration. Note that CAS latency plus additive latency must be at least 3 cycles for ODT_RD_CFG to be enabled.</p> <ul style="list-style-type: none"> <li>000b - Never assert ODT for reads</li> <li>001b - Assert ODT only during reads to CS<sub>a</sub></li> <li>010b - Assert ODT only during reads to other chip selects</li> <li>011b - Assert ODT only during reads to other DIMM modules. It is assumed that CS0 and CS1 are on the same DIMM module, whereas CS2 and CS3 are on a separate DIMM module.</li> <li>100b - Assert ODT for all reads</li> <li>101b - Assert ODT only during transactions to same DIMM</li> <li>110b - Assert ODT only during transactions to own CS and other DIMM.</li> <li>111b - Assert ODT only during transactions to other CS in same DIMM.</li> </ul>
12 —	Reserved
13-15 ODT_WR_CFG	<p>On-Die Termination Write Config.</p> <p>ODT for writes configuration. Note that write latency plus additive latency must be at least 3 cycles for ODT_WR_CFG to be enabled.</p> <ul style="list-style-type: none"> <li>000b - Never assert ODT for writes</li> <li>001b - Assert ODT only during writes to CS<sub>n</sub></li> <li>010b - Assert ODT only during writes to other chip selects</li> <li>011b - Assert ODT only during writes to other DIMM modules. It is assumed that CS0 and CS1 are on the same DIMM module, whereas CS2 and CS3 are on a separate DIMM module.</li> <li>100b - Assert ODT for all writes</li> <li>101b - Assert ODT only during transactions to same DIMM</li> <li>110b - Assert ODT only during transactions to own CS and other DIMM.</li> <li>111b - Assert ODT only during transactions to other CS in same DIMM.</li> </ul>
16-17 BA_BITS_CS	<p>Bank Address Bits.</p> <p>Number of bank bits for SDRAM on chip select<sub>n</sub>. These bits correspond to the sub-bank bits driven on MBAn. Note that if DDR4 is used, this must be set to 00, as 8 sub-banks are not supported when also using bank groups.</p> <ul style="list-style-type: none"> <li>00b - 2 logical bank bits</li> <li>01b - 3 logical bank bits</li> <li>10b - Reserved</li> <li>11b - Reserved</li> </ul>
18-20 —	Reserved
21-23 ROW_BITS_CS	<p>Row Bits.</p> <p>Number of row bits for SDRAM on chip select<sub>n</sub>.</p> <ul style="list-style-type: none"> <li>000b - 12 row bits</li> <li>001b - 13 row bits</li> <li>010b - 14 row bits</li> <li>011b - 15 row bits</li> </ul>

Table continues on the next page...

## DDR register descriptions

Field	Function
	100b - 16 row bits 101b - 17 row bits 110b - 18 row bits 111b - Reserved
24-25 —	Reserved
26-27 BG_BITS_CS	Bank Group Bits. Number of bank group bits for SDRAM on chip selectn. In addition, it is illegal to use BA_BITS_CS_n equal to 01 if DDR4 bank groups are used. 00b - 0 bank group bits 01b - 1 bank group bit 10b - 2 bank group bits 11b - Reserved
28 —	Reserved
29-31 COL_BITS_CS	Column Bits. Number of column bits for SDRAM on chip selectn. The decoding is as follows: 000b - 8 column bits 001b - 9 column bits 010b - 10 column bits 011b - 11 column bits 100b - Reserved 101b - Reserved 110b - Reserved 111b - Reserved

## 18.4.4 DDR SDRAM timing configuration 3 (TIMING\_CFG\_3)

### 18.4.4.1 Offset

Register	Offset
TIMING_CFG_3	100h

### 18.4.4.2 Function

DDR SDRAM timing configuration register 3 sets the extended refresh recovery time, which is combined with TIMING\_CFG\_1[REFREC] to determine the full refresh recovery time.

### 18.4.4.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			EXT_PRETOACT	Reserved		EXT_ACTTOPRE		Reserved		EXT_ACTTOWRW		EXT_REFRE			
W													C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			EXT_CASLAT	Reserved	EXT_ADD_LAT	Reserved	EXT_WRRE	Reserved					CNTL_ADJ		
W								C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.4.4 Fields

Field	Function
0-2	Reserved
—	
3	Extended Precharge to Activate.
EXT_PRETOACT	Extended precharge-to-activate interval ( $t_{RP}$ ). Determines the number of clock cycles from a precharge command until an activate or refresh command is allowed. This field is concatenated with TIMING_CFG_1[PRETOACT] to obtain a 5-bit value for the total precharge to activate time. 0b - 0 clocks 1b - 16 clocks
4-5	Reserved
—	
6-7	Extended Activate to Precharge.
EXT_ACTTOPRE	Extended Activate to precharge interval ( $t_{RAS}$ ). Determines the number of clock cycles from an activate command until a precharge command is allowed. This field is concatenated with TIMING_CFG_1[ACTTOPRE] to obtain a 6-bit value for the total activate to precharge. Note that a 6-bit value of 00_0000 is the same as a 6-bit value of 01_0000. Both values represent 16 cycles. 00b - 0 clocks 01b - 16 clocks 10b - 32 clocks 11b - 48 clocks
8	Reserved
—	

Table continues on the next page...

## DDR register descriptions

Field	Function
9 EXT_ACTTOR_W	Extended Activate to Read/Write. Extended activate to read/write interval for SDRAM ( $t_{RCD}$ ). Controls the number of clock cycles from an activate command until a read or write command is allowed. This field is concatenated with TIMING_CFG_1[ACTTORW] to obtain a 5-bit value for the total activate to read/write time.
10-15 EXT_REFREC	Extended Refresh Recovery. Extended refresh recovery time ( $t_{RFC}$ ). Controls the number of clock cycles from a refresh command until an activate command is allowed. This field is concatenated with TIMING_CFG_1[REFREC] to obtain a 10-bit value for the total refresh recovery. Note that hardware adds an additional 8 clock cycles to the final, 10-bit value of the refresh recovery, such that $t_{RFC}$ is calculated as follows: $t_{RFC} = \{EXT\_REFREC \parallel REFREC\} + 8$ . (Settings greater than 101111b are reserved.) All values of less than 0b110000 are legal; in this case the number of cycles is [setting] $\times$ 16. Sample values are shown below. Settings of 0b110000 or greater are reserved. 000000b - 0 clocks 000001b - 16 clocks 000010b - 32 clocks 101110b - 736 clocks 101111b - 752 clocks 110000-111111b - Reserved
16-17 —	Reserved
18-19 EXT_CASLAT	Extended CAS Latency. Number of clock cycles between registration of a READ command by the SDRAM and the availability of the first output data. If a READ command is registered at clock edge $n$ and the latency is $m$ clocks, data is available nominally coincident with clock edge $n + m$ . This field is concatenated with TIMING_CFG_1[CASLAT] to obtain a 5-bit value for the total CAS latency. Note that the value of this field is added to the programmed value in TIMING_CFG_1[CASLAT]. The largest total CAS latency supported is 20 clocks. 00b - 0 clocks 01b - 8 clocks 10b - 16 clocks 11b - Reserved
20 —	Reserved
21 EXT_ADD_LAT	Extended Additive Latency. The additive latency must be set to a value less than TIMING_CFG_1[ACTTORW]. Note that the value of this field is added to the programmed value in TIMING_CFG_2[ADD_LAT]. The largest total additive latency supported is 19 clocks. 0b - 0 clocks 1b - 16 clocks
22 —	Reserved
23 EXT_WRREC	Extended Write Recovery. Extended last data to precharge minimum interval ( $t_{WR}$ ). Determines the number of clock cycles from the last data associated with a write command until a precharge command is allowed. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to ( $t_{WR} + 2$ cycles). This field is concatenated with TIMING_CFG_1[WRREC] to obtain a 5-bit value for the total write recovery time. 0b - 0 clocks

Table continues on the next page...

Field	Function
	1b - 16 clocks
24-28 —	Reserved
29-31 CNTL_ADJ	<p>Control Adjust.</p> <p>Controls the amount of delay to add to the lightly loaded control signals with respect to all other DRAM address and command signals. The signals affected by this field are MODTn, MCSn_B, and MCKEn.</p> <p>000b - MODTn, MCSn_B, and MCKEn will be launched aligned with the other DRAM address and control signals.</p> <p>001b - MODTn, MCSn_B, and MCKEn will be launched 1/4 DDR clock cycle later than the other DRAM address and control signals.</p> <p>010b - MODTn, MCSn_B, and MCKEn will be launched 1/2 DDR clock cycle later than the other DRAM address and control signals.</p> <p>011b - MODTn, MCSn_B, and MCKEn will be launched 3/4 DDR clock cycle later than the other DRAM address and control signals.</p> <p>100b - MODTn, MCSn_B, and MCKEn will be launched 1 DDR clock cycle later than the other DRAM address and control signals.</p> <p>101b - MODTn, MCSn_B, and MCKEn will be launched 5/4 DDR clock cycles later than the other DRAM address and control signals.</p> <p>110b - Reserved</p> <p>111b - Reserved</p>

## 18.4.5 DDR SDRAM timing configuration 0 (TIMING\_CFG\_0)

### 18.4.5.1 Offset

Register	Offset
TIMING_CFG_0	104h

### 18.4.5.2 Function

DDR SDRAM timing configuration register 0 sets the number of clock cycles between various SDRAM control commands.

### 18.4.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RWT		WRT		RRT		WWT		ACT_PD_EXIT			PRE_PD_EXIT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	EXT_PRE_PD_EXIT															
W			Reserved													MRS_CYC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

### 18.4.5.4 Fields

Field	Function
0-1 RWT	<p>Read-to-write Turnaround.</p> <p>Specifies how many extra cycles will be added between a read to write turnaround. If 0 clocks is chosen, then the DDR controller will use a fixed number based on the CAS latency and write latency. Choosing a value other than 0 adds extra cycles past this default calculation. As a default the DDR controller will determine the read-to-write turnaround as <math>CL - WL + BL/2 + 2</math>. In this equation, CL is the CAS latency rounded up to the next integer, WL is the programmed write latency, and BL is the burst length. This field is concatenated with TIMING_CFG_4[EXT_RWT] to obtain a 4-bit value for the total read-to-write turnaround</p> <ul style="list-style-type: none"> <li>00b - 0 clocks</li> <li>01b - 1 clock</li> <li>10b - 2 clocks</li> <li>11b - 3 clocks</li> </ul>
2-3 WRT	<p>Write-to-read Turnaround.</p> <p>Specifies how many extra cycles will be added between a write to read turnaround. If 0 clocks is chosen, then the DDR controller will use a fixed number based on the read latency and write latency. Choosing a value other than 0 adds extra cycles past this default calculation. As a default, the DDR controller will determine the write-to-read turnaround as <math>WL - CL + BL/2 + 1</math>. In this equation, CL is the CAS latency rounded down to the next integer, WL is the programmed write latency, and BL is the burst length. This field is concatenated with TIMING_CFG_4[EXT_WRT] to obtain a 3-bit value for the total write-to-read turnaround</p> <ul style="list-style-type: none"> <li>00b - 0 clocks</li> <li>01b - 1 clock</li> <li>10b - 2 clocks</li> <li>11b - 3 clocks</li> </ul>
4-5 RRT	Read-to-read Turnaround.

Table continues on the next page...

Field	Function
	<p>Specifies how many extra cycles will be added between reads to different chip selects. As a default, 3 cycles will be required between read commands to different chip selects. Extra cycles may be added with this field. Note: If 8-beat bursts are enabled, then 5 cycles will be the default. This field is concatenated with TIMING_CFG_4[EXT_RRT] to obtain a 3-bit value for the total read-to-read turnaround.</p> <p>00b - 0 clocks 01b - 1 clock 10b - 2 clocks 11b - 3 clocks</p>
6-7 WWT	<p>Write-to-write Turnaround.</p> <p>Specifies how many extra cycles will be added between writes to different chip selects. As a default, 2 cycles will be required between write commands to different chip selects. Extra cycles may be added with this field. Note: If 8-beat bursts are enabled, then 4 cycles will be the default. This field is concatenated with TIMING_CFG_4[EXT_WWT] to obtain a 3-bit value for the total write-to-write turnaround</p> <p>00b - 0 clocks 01b - 1 clock 10b - 2 clocks 11b - 3 clocks</p>
8-11 ACT_PD_EXIT	<p>Active Powerdown Exit.</p> <p>Active powerdown exit timing (<math>t_{XP}</math>). Specifies how many clock cycles to wait after exiting active powerdown before issuing any command.</p> <p>0000b - Reserved 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks</p>
12-15 PRE_PD_EXIT	<p>Precharge Powerdown Exit.</p> <p>Precharge powerdown exit timing (<math>t_{XP}</math>). Specifies how many clock cycles to wait after exiting precharge powerdown before issuing any command. This field is concatenated with TIMING_CFG_0[EXT_PRE_PD_EXIT] to obtain a 6-bit value for the total precharge powerdown exit timing.</p> <p>0000b - Reserved 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks</p>

Table continues on the next page...

## DDR register descriptions

Field	Function
	1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
16-17 EXT_PRE_PD_EXIT	Extended Precharge Powerdown Exit.  Extended precharge powerdown exit timing ( $t_{XP}$ ). Specifies how many clock cycles to wait after exiting precharge powerdown before issuing any command. Note the decoding for this field is not a straight decode. This field is concatenated with TIMING_CFG_0[PRE_PD_EXIT] to obtain a 6-bit value for the total precharge powerdown exit timing.  00b - 0 clocks 01b - 16 clocks 10b - 32 clocks 11b - 48 clocks
18-26 —	
27-31 MRS_CYC	Mode Register Set Cycle Time.  Mode register set cycle time ( $t_{MRD}$ , $t_{MOD}$ ). Specifies the number of cycles that must pass after a Mode Register Set command until any other command. This should be set to the greater of $t_{MRD}$ and $t_{MOD}$ . If command/address latency (CAL) mode is used for DDR4, then this field should be set to the greater of $t_{MRD\_CAL}$ and $t_{MOD\_CAL}$ . In addition, this field should be programmed higher than TIMING_CFG_7[CS_TO_CMD] if using CAL mode.  00000b - Reserved 00001b - 1 clock 00010b - 2 clocks 00011b - 3 clocks 00100b - 4 clocks 00101b - 5 clocks 00110b - 6 clocks 00111b - 7 clocks 01000b - 8 clocks 01001b - 9 clocks 01010b - 10 clocks 01011b - 11 clocks 01100b - 12 clocks 01101b - 13 clocks 01110b - 14 clocks 01111b - 15 clocks 10000b - 16 clocks 10001b - 17 clocks 10010b - 18 clocks 10011b - 19 clocks 10100b - 20 clocks 10101b - 21 clocks 10110b - 22 clocks 10111b - 23 clocks 11000b - 24 clocks 11001b - 25 clocks 11010b - 26 clocks 11011b - 27 clocks 11100b - 28 clocks 11101b - 29 clocks 11110b - 30 clocks 11111b - 31 clocks

## 18.4.6 DDR SDRAM timing configuration 1 (TIMING\_CFG\_1)

### 18.4.6.1 Offset

Register	Offset
TIMING_CFG_1	108h

### 18.4.6.2 Function

DDR SDRAM timing configuration register 1 sets the number of clock cycles between various SDRAM control commands.

### 18.4.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<b>PRETOACT</b>				<b>ACTTOPRE</b>				<b>ACTTOWR</b>				<b>CASLAT</b>			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	<b>REFREC</b>				<b>WRREC</b>				<b>ACTTOACT</b>				<b>WRTORD</b>			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.6.4 Fields

Field	Function
0-3 PRETOACT	Precharge-to-Activate. Precharge-to-activate interval ( $t_{RP}$ ). Determines the number of clock cycles from a precharge command until an activate or refresh command is allowed. This field is concatenated with TIMING_CFG_3[EXT_PRETOACT] to obtain a 5-bit value for the total precharge to activate time. 0000b - Reserved 0001b - 1 clock 0010b - 2 clocks

Table continues on the next page...

## DDR register descriptions

Field	Function
	0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
4-7 ACTTOPRE	Activate-to-Precharge.  Activate to precharge interval ( $t_{RAS}$ ). Determines the number of clock cycles from an activate command until a precharge command is allowed. This field is concatenated with TIMING_CFG_3[EXT_ACTTOPRE] to obtain a 6-bit value for the total activate to precharge time. Note that the decode of 0000-0011 is equal to 16-19 clocks when TIMING_CFG_3[EXT_ACTTOPRE] = 00, but it is equal to 0-3 clocks when TIMING_CFG_3[EXT_ACTTOPRE] = 01, 10, or 11.  0000b - 16 clocks 0001b - 17 clocks 0010b - 18 clocks 0011b - 19 clocks 0100b - 4 clock 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
8-11 ACTTOWR	Activate-to-Read/Write.  Activate to read/write interval for SDRAM ( $t_{RCD}$ ). Controls the number of clock cycles from an activate command until a read or write command is allowed. This field is concatenated with TIMING_CFG_3[EXT_ACTTOWR] to obtain a 5-bit value for the total activate to read/write time.  0000b - Reserved 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks

Table continues on the next page...

Field	Function
	1111b - 15 clocks
12-14 CASLAT	<p>CAS Latency.</p> <p>Number of clock cycles between registration of a READ command by the SDRAM and the availability of the first output data. If a READ command is registered at clock edge <math>n</math> and the latency is <math>m</math> clocks, data is available nominally coincident with clock edge <math>n + m</math>. This field is concatenated with TIMING_CFG_3[EXT_CASLAT] to obtain a 5-bit value for the total CAS latency. This value must be programmed at initialization as described in <a href="#">DDR SDRAM control configuration 2 (DDR_SDRAM_CFG_2)</a>) Note that the largest total CAS latency supported is 20 clocks.</p> <ul style="list-style-type: none"> <li>000b - 1 clock</li> <li>001b - 2 clocks</li> <li>010b - 3 clocks</li> <li>011b - 4 clocks</li> <li>100b - 5 clocks</li> <li>101b - 6 clocks</li> <li>110b - 7 clocks</li> <li>111b - 8 clocks</li> </ul>
15 RSRVD	<p>Reserved</p> <p>This bit is reserved, but it is readable and writeable.</p>
16-19 REFREC	<p>Refresh recovery.</p> <p>Refresh recovery time (<math>t_{RFC}</math>). Controls the number of clock cycles from a refresh command until an activate command is allowed. This field is concatenated with TIMING_CFG_3[EXTREFREC] to obtain a 10-bit value for the total refresh recovery. Note that hardware adds an additional 8 clock cycles to the final, 10-bit value of the refresh recovery, such that <math>t_{RFC}</math> is calculated as follows: <math>t_{RFC} = \{EXT\_REFREC \parallel REFREC\} + 8</math>.</p> <ul style="list-style-type: none"> <li>0000b - 8 clocks</li> <li>0001b - 9 clocks</li> <li>0010b - 10 clocks</li> <li>0011b - 11 clocks</li> <li>1111b - 23 clocks</li> </ul>
20-23 WRREC	<p>Write recovery.</p> <p>Last data to precharge minimum interval (<math>t_{WR}</math>). Determines the number of clock cycles from the last data associated with a write command until a precharge command is allowed. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to (<math>t_{WR} + 2</math> cycles). This field is concatenated with TIMING_CFG_3[EXT_WRREC] to obtain a 5-bit value for the total write recovery time.</p> <ul style="list-style-type: none"> <li>0000b - Reserved</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
24-27	Activate-to-activate.

Table continues on the next page...

## DDR register descriptions

Field	Function
ACTTOACT	Activate-to-activate interval ( $t_{RRD}$ ). Number of clock cycles from an activate command until another activate command is allowed for a different logical bank in the same physical bank (chip select). 0000b - Reserved 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
28-31	Write-to-read.
WRTORD	Last write data pair to read command issue interval ( $t_{WTR}$ ). Number of clock cycles between the last write data pair and the subsequent read command to the same physical bank. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to ( $t_{WTR} + 2$ cycles). 0000b - Reserved 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks

## 18.4.7 DDR SDRAM timing configuration 2 (TIMING\_CFG\_2)

### 18.4.7.1 Offset

Register	Offset
TIMING_CFG_2	10Ch

### 18.4.7.2 Function

DDR SDRAM timing configuration 2 sets the clock delay to data for writes.

### 18.4.7.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<i>ADD_LAT</i>															
W					<i>Reserved</i>				<i>WR_LAT</i>					<i>EXT_WR_LAT</i>		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	<i>RD_TO_PR</i>			<i>WR_DATA_DELAY</i>					<i>CKE_PL</i>				<i>FOUR_ACT</i>			
W	E								S							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.7.4 Fields

Field	Function
0-3 ADD_LAT	Additive Latency.  The additive latency must be set to a value less than TIMING_CFG_1[ACTTOWR]. This field is added to TIMING_CFG_3[EXT_ADD_LAT]. The maximum total additive latency supported is 19 clocks.  0000b - 0 clocks 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks

Table continues on the next page...

## DDR register descriptions

Field	Function
	1111b - Reserved
4-8 —	Reserved
9-12 WR_LAT	<p>Write Latency.</p> <p>Note that the total write latency is equal to WR_LAT + ADD_LAT. Note that the total write latency must be at least 6 cycles if using unbuffered DIMMs in 1T timing mode. Note that this field is added to TIMING_CFG_2[EXT_WR_LAT]. The maximum write latency supported before adding the additive latency is 18 clocks. Note that values of 0b0000 to 0b0101 are reserved unless TIMING_CFG_2[EXT_WR_LAT] is programmed to a non-zero value.</p> <ul style="list-style-type: none"> <li>0000b - 0 clocks</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
13 EXT_WR_LAT	<p>Extended Write Latency.</p> <p>Note that the value of this field is added to the programmed value in TIMING_CFG_2[WR_LAT]. The largest total write latency supported before adding the additive latency is 18 clocks.</p> <ul style="list-style-type: none"> <li>0b - 0 clocks</li> <li>1b - 16 clocks</li> </ul>
14 —	Reserved
15-18 RD_TO_PRE	<p>Read-to-Precharge.</p> <p>Read to precharge (<math>t_{RTP}</math>). If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to (<math>t_{RTP} + 2</math> cycles).</p> <ul style="list-style-type: none"> <li>0000b - Reserved</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>

Table continues on the next page...

Field	Function
19-22 WR_DATA_DEL_AY	<p>Write Data Delay.</p> <p>Write command to write data strobe timing adjustment. Controls the amount of delay applied to the data and data strobes for writes. See <a href="#">DDR SDRAM Write Timing Adjustments</a> for details. The write preamble will be driven high for 1/2 DRAM cycle, and then it will be driven low for 1/2 DRAM cycle.</p> <ul style="list-style-type: none"> <li>0000b - 0 clock delay</li> <li>0001b - 2 clock delay</li> <li>0010b - 1/4 clock delay</li> <li>0011b - 9/4 clock delay</li> <li>0100b - 1/2 clock delay</li> <li>0101b - 5/2 clock delay</li> <li>0110b - 3/4 clock delay</li> <li>0111b - Reserved</li> <li>1000b - 1 clock delay</li> <li>1001b - Reserved</li> <li>1010b - 5/4 clock delay</li> <li>1011b - Reserved</li> <li>1100b - 3/2 clock delay</li> <li>1101b - Reserved</li> <li>1110b - 7/4 clock delay</li> <li>1111b - Reserved</li> </ul>
23-25 CKE_PLS	<p>CKE Pulse.</p> <p>Minimum CKE pulse width (<math>t_{CKE}</math>). This field is concatenated with TIMING_CFG_3[EXT_CKE_PLS] to obtain a 4-bit value for the total minimum CKE pulse width.</p> <ul style="list-style-type: none"> <li>000b - 8 clocks</li> <li>001b - 1 clock</li> <li>010b - 2 clocks</li> <li>011b - 3 clocks</li> <li>100b - 4 clocks</li> <li>101b - 5 clocks</li> <li>110b - 6 clocks</li> <li>111b - 7 clocks</li> </ul>
26-31 FOUR_ACT	<p>Four Activate.</p> <p>Window for four activates (<math>t_{FAW}</math>).</p> <p><b>NOTE:</b> If <math>t_{FAW}</math> requires FOUR_ACT to be set higher than the supported values, then TIMING_CFG_1[ACTTOACT] should be programmed to avoid a <math>t_{FAW}</math> violation. This can be done by programming -&gt; TIMING_CFG_1[ACTTOACT] = rounded_up[max(tRRD{_S}, tFAW/4)].</p> <ul style="list-style-type: none"> <li>000000b - Reserved</li> <li>000001b - 1 cycle</li> <li>000010b - 2 cycles</li> <li>000011b - 3 cycles</li> <li>000100b - 4 cycles</li> <li>011111b - 31 cycles</li> <li>100000b - 32 cycles</li> </ul>

## 18.4.8 DDR SDRAM control configuration (DDR\_SDRAM\_CFG)

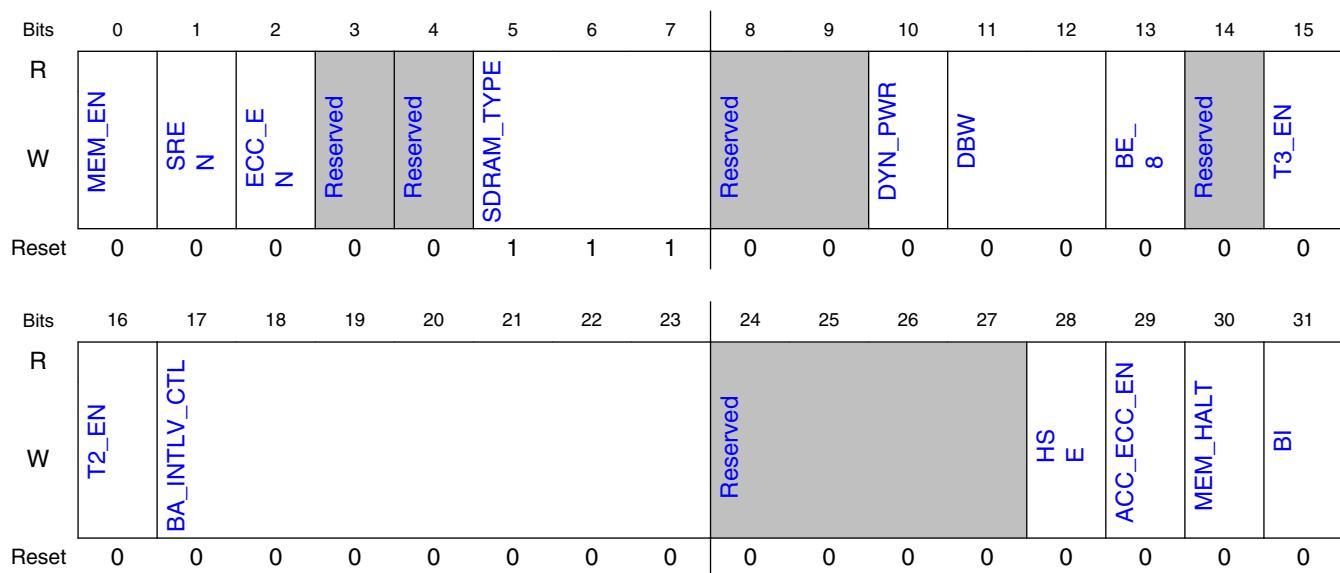
### 18.4.8.1 Offset

Register	Offset
DDR_SDRAM_CFG	110h

### 18.4.8.2 Function

The DDR SDRAM control configuration register enables the interface logic and specifies certain operating features such as self refreshing, error checking and correcting, and dynamic power management.

### 18.4.8.3 Diagram



### 18.4.8.4 Fields

Field	Function
0 MEM_EN	Memory Controller Enable. 0b - SDRAM interface logic is disabled. 1b - SDRAM interface logic is enabled. Must not be set until all other memory configuration parameters have been appropriately configured by initialization code.
1 SREN	Self Refresh Enable.

Table continues on the next page...

Field	Function
	0b - SDRAM self refresh is disabled during sleep. Whenever self-refresh is disabled, the system is responsible for preserving the integrity of SDRAM during sleep. 1b - SDRAM self refresh is enabled during sleep.
2 ECC_EN	ECC Enable.  Note that uncorrectable read errors may cause an interrupt.  <b>NOTE:</b> If this bit is set to 1, DDR_SDRAM_CFG[ACC_ECC_EN] must be set to 1 as well. 0b - No ECC errors are reported. No ECC interrupts are generated. 1b - ECC is enabled.
3 —	Reserved
4 —	Reserved
5-7 SDRAM_TYPE	SDRAM Type.  Type of SDRAM device to be used. This field will be used when issuing the automatic hardware initialization sequence to DRAM via Mode Register Set and Extended Mode Register Set commands. Default value is 111.  000-100 Reserved 101 DDR4 SDRAM 110 Reserved 111 DDR3-type SDRAM
8-9 —	Reserved
10 DYN_PWR	Dynamic Power Management. 0b - Dynamic power management mode is disabled. 1b - Dynamic power management mode is enabled. If there is no ongoing memory activity, the SDRAM CKE signal is negated.
11-12 DBW	DRAM Data Bus Width. 00b - Reserved 01b - 32-bit bus is used. 10b - 16-bit bus is used. 11b - Reserved
13 BE_8	8-Beat Burst Enable. 0b - 4-beat bursts are used on the DRAM interface. This is only supported if DDR_SDRAM_CFG_2[OBC_CFG] is also set. 1b - 8-beat bursts are used on the DRAM interface.
14 —	Reserved
15 T3_EN	3T Timing Enable.  This field cannot be set if DDR_SDRAM_CFG[T2_EN] is also set. This field cannot be used with a 32-bit bus or a 16-bit bus if 4-beat bursts are used.  <b>NOTE:</b> 3T timing may not be used with 4-beat bursts, unless DDR_SDRAM_CFG_2[OBC_CFG] is set. 0b - 1T timing is enabled if T2_EN is cleared. The DRAM command/address are held for only 1 cycle on the DRAM bus. 1b - 3T timing is enabled. The DRAM command/address are held for 3 full cycles on the DRAM bus for every DRAM transaction. However, the chip select is only held for the third cycle.
16	2T Timing Enable.

Table continues on the next page...

## DDR register descriptions

Field	Function
T2_EN	<p>This field should not be set if DDR_SDRAM_CFG[T3_EN] is set.</p> <p>0b - 1T timing is enabled if T3_EN is cleared. The DRAM command/address are held for only 1 cycle on the DRAM bus.</p> <p>1b - 2T timing is enabled. The DRAM command/address are held for 2 full cycles on the DRAM bus for every DRAM transaction. However, the chip select is only held for the second cycle.</p>
17-23 BA_INTLV_CTL	<p>Bank (chip select) interleaving control.</p> <p>Set this field only when using bank interleaving.</p> <p>('x' denotes a don't care bit value. All unlisted field values are reserved.)</p> <ul style="list-style-type: none"> <li>0000000 No external memory banks are interleaved</li> <li>1000000 External memory banks 0 and 1 are interleaved</li> <li>0100000 External memory banks 2 and 3 are interleaved</li> <li>1100000 External memory banks 0 and 1 are interleaved together and banks 2 and 3 are interleaved together</li> <li>xx00100 External memory banks 0 through 3 are all interleaved together</li> </ul>
24-27 —	Reserved
28 HSE	<p>Half-Strength Enable.</p> <p>Sets I/O driver impedance to calibrate to half strength. This calibrated impedance will be used by the MDIC, address/command, data, and clock impedance values, but only if automatic hardware calibration is enabled and the corresponding group's software override is disabled in the DDR control driver register(s) described in <a href="#">DDR Control Driver Register 1 (DDRCDR_1)</a> and <a href="#">DDR Control Driver Register 2 (DDRCDR_2)</a></p> <p>0b - I/O driver impedance will be calibrated to full strength. 1b - I/O driver impedance will be calibrated to half strength.</p>
29 ACC_ECC_EN	<p>Accumulated ECC enable.</p> <p>This can be used to save ECC pins/wires when using a DDR data bus width smaller than 64-bits. In this mode, the 8-bit ECC code will be transmitted/received across several beats, and it will be used to check 64-bits of data once 8-bits of ECC are accumulated. Note that using this mode guarantees that all single-bit ECC errors are corrected and detected for every 64-bits of data, and every 2-bit ECC error is detected for 64-bits of data. Unused ECC bits are driven high by the memory controller during write bursts if ACC_ECC_EN is set.</p> <p><b>NOTE:</b> This bit must be set to 1 when ECC is enabled.</p> <ul style="list-style-type: none"> <li>0b - Accumulated ECC is disabled</li> <li>1b - Accumulated ECC is enabled</li> </ul>
30 MEM_HALT	<p>Memory Controller Halt.</p> <p>DDR memory controller halt. When this bit is set, the memory controller will not accept any new data read/write transactions to DDR SDRAM until the bit is cleared again. This can be used when bypassing initialization and forcing MODE REGISTER SET commands through software.</p> <p>0b - DDR controller will accept new transactions. 1b - DDR controller will finish any remaining transactions, and then it will remain halted until this bit is cleared by software.</p>
31 BI	<p>Bypass Initialization.</p> <p>See <a href="#">DDR training initialization address (DDR_INIT_ADDR)</a> for details on avoiding ECC errors in this mode.</p> <p>0b - DDR controller will cycle through initialization routine based on SDRAM_TYPE</p>

Field	Function
	1b - Initialization routine will be bypassed. Software is responsible for initializing memory through DDR_SDRAM_MD_CNTL register. If software is initializing memory, then the MEM_HALT bit can be set to prevent the DDR controller from issuing transactions during the initialization sequence. Note that the DDR controller will not issue a DLL reset to the DRAMs when bypassing the initialization routine, regardless of the value of DDR_SDRAM_CFG[DLL_RST_DIS]. If a DLL reset is required, then the controller should be forced to enter and exit self refresh after the controller is enabled.

## 18.4.9 DDR SDRAM control configuration 2 (DDR\_SDRAM\_CFG\_2)

### 18.4.9.1 Offset

Register	Offset
DDR_SDRAM_CFG_2	114h

### 18.4.9.2 Function

The DDR SDRAM control configuration register 2 provides more control configuration for the DDR controller.

### 18.4.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	FRC_S R	Reserved	Reserved							ODT_CFG						
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	NUM_PR				DDR_SLOW	Reserved	QD_EN	UNQ_MRS_EN	Reserved	OBC_CFG	AP_EN	D_INIT	SPARE_CNF G	Reserved	CD_DIS	MD_EN
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.9.4 Fields

Field	Function
0 FRC_SR	Force Self Refresh. 0b - DDR controller will operate in normal mode. 1b - DDR controller will enter self-refresh mode.
1 —	Reserved
2-8 —	Reserved
9-10 ODT_CFG	ODT configuration. This field defines how ODT will be driven to the on-chip IOs. See <a href="#">DDR Control Driver Register 1 (DDRC DR_1)</a> and <a href="#">DDR Control Driver Register 2 (DDRCDR_2)</a> which define the termination value that will be used. 00b - Never assert ODT to internal IOs 01b - Reserved 10b - Assert ODT to internal IOs only during reads to DRAM 11b - Reserved
11-15 —	Reserved
16-19 NUM_PR	Number of posted refreshes. This will determine how many posted refreshes, if any, can be issued at one time. Note that if posted refreshes are used, then this field, along with DDR_SDRAM_INTERVAL[REFINT], must be programmed such that the maximum $t_{ras}$ specification cannot be violated. Patterns not shown are reserved. 0000b - Reserved 0001b - 1 refresh will be issued at a time 0010b - 2 refreshes will be issued at a time 0011b - 3 refreshes will be issued at a time 1000b - 8 refreshes will be issued at a time
20 DDR_SLOW	DDR Slow Frequency. Indicates to the controller if it will be run at a lower frequency. 0b - The DDR controller will be run at data rates of 1250 MT/s or higher. 1b - The DDR controller will be run at data rates of less than 1250 MT/s.
21 —	Reserved
22 QD_EN	Quad-Rank Enable. Determines if a quad-ranked DIMM is used. This bit should also be set if quad-stacked discrete memory chips are used. 0b - Quad-ranked DIMMs are not used. 1b - Quad-ranked DIMMs are used.
23 UNQ_MRS_EN	Unique MRS Enable.

Table continues on the next page...

Field	Function
	Determines if the DDR_SDRAM_MODE_{3:8} and DDR_SDRAM_MODE_{11:16} registers will be used when initializing the memories for chip selects 1, 2, and 3. These can be used to provide unique values to the Mode Registers of the DRAM to allow different termination values for each rank.
24 —	
25 OBC_CFG	<p>On-The-Fly Burst Chop Configuration.</p> <p>Determines if on-the-fly Burst Chop will be used. If on-the-fly Burst Chop mode is not used, then 8-beat burst mode should be used. DDR_SDRAM_CFG[BE_8] should be cleared for on-the-fly Burst Chop mode.</p> <p>0b - On-the-fly Burst Chop mode is disabled. Fixed burst lengths as defined in DDR_SDRAM_CFG[BE_8] are used.            1b - On-the-fly Burst Chop mode will be used. DDR_SDRAM_CFG[BE_8] should be cleared for on-the-fly Burst Chop mode. DDR_SDRAM_CFG[DBW] should also be programmed for 64-bit bus or 32-bit bus.</p>
26 AP_EN	<p>Address Parity Enable.</p> <p>Determines if address parity will be generated and checked for the address and control signals . If address parity is used, the MAPAR_OUT and MAPAR_ERR_B pins will be used to drive the parity bit and to receive errors from the open-drain parity error signal. Even parity will be used, and parity will be generated for the MA[15:0], MBA[2:0], MRAS_B, MCAS_B, MWE_B signals. Parity will not be generated for the MODTn, MCSn_B, and MCKEn signals. Note that address parity should not be used for non-zero values of TIMING_CFG_3[CNTL_ADJ].</p> <p>0b - Address parity will not be used            1b - Address parity will be used</p>
27 D_INIT	<p>DRAM data initialization.</p> <p>This bit is set by software, and it is cleared by hardware. If software sets this bit before the memory controller is enabled, the controller will automatically initialize DRAM after it is enabled. This bit will be automatically cleared by hardware once the initialization is completed. This data initialization bit should only be set when the controller is idle.</p> <p>0b - There is not data initialization in progress, and no data initialization is scheduled            1b - The memory controller will initialize memory once it is enabled. This bit will remain asserted until the initialization is complete. The value in DDR_DATA_INIT register will be used to initialize memory.</p>
28 SPARE_CFG	Spare Config Bits. This field is currently unused.
29 —	Reserved
30 CD_DIS	<p>Corrupted Data Disable.</p> <p>If this bit is set, then the corrupted data feature will be disabled. When the corrupted data feature is enabled, the DDR controller will invert the generated ECC code for any beat of data which is known to have corrupted data. When a read to the corrupted data is later generated, the <i>ERR_DETECT/CDE</i> error will be set if error reporting is enabled.</p> <p>0b - Corrupted data is enabled            1b - Corrupted data is disabled</p>
31 MD_EN	<p>Mirrored DIMM Enable.</p> <p>Some DIMMs will be mirrored, where certain MA and MBA pins are mirrored on one side of the DIMM. When this bit is set, the controller will know to swap these signals before transmitting to the DRAM. The controller will assume that CS1 and CS3 are the 'mirrored' ranks of memory.</p> <p>The following signals are mirrored for DDR3:</p>

Field	Function
	<ul style="list-style-type: none"> <li>• MBA[0] vs. MBA[1]</li> <li>• MA[3] vs. MA[4]</li> <li>• MA[5] vs. MA[6]</li> <li>• MA[7] vs. MA[8]</li> </ul> <p>The following signals are mirrored for DDR4:</p> <ul style="list-style-type: none"> <li>• MBA[0] vs. MBA[1]</li> <li>• MA[3] vs. MA[4]</li> <li>• MA[5] vs. MA[6]</li> <li>• MA[7] vs. MA[8]</li> <li>• MA[11] vs. MA[13]</li> <li>• MBG0 vs. MBG1 (if 2 bank group bits are used)</li> </ul> <p>Note that MBG0 is the same as MBA[2] from the DDR controller. If using mirrored DIMMs for DDR4 mode, then CS0 and CS1 must be programmed to use the same number of bank group bits in CSa_CONFIG[BG_BITS_CSa]. CS2 and CS3 must also be programmed to use the same number of bank group bits.</p> <p>0b - Mirrored DIMMs are not used 1b - Mirrored DIMMs are used</p>

## 18.4.10 DDR SDRAM mode configuration (DDR\_SDRAM\_MODE)

### 18.4.10.1 Offset

Register	Offset
DDR_SDRAM_MODE	118h

### 18.4.10.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers.

### 18.4.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SDMODE															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.10.4 Fields

Field	Function
0-15 ESDMODE	<p>Extended SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM extended mode register. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb of ESDMODE, which, in the big-endian convention shown here, corresponds to ESDMODE[15]. The msb of the SDRAM extended mode register value must be stored at ESDMODE[0]. The value programmed into this field is also used for writing MR1 during write leveling for DDR3 memory types, although the bits specifically related to the write leveling scheme are handled automatically by the DDR controller. Even if DDR_SDRAM_CFG[B1] is set, this field is still used during write leveling. The write leveling enable bit should be cleared by software in this field.</p>
16-31 SDMODE	<p>SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM mode register. The range of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of SDMODE, which, in the big-endian convention, corresponds to SDMODE[15]. The msb of the SDRAM mode register value must be stored at SDMODE[0]. Because the memory controller forces SDMODE[7] to certain values depending on the state of the initialization sequence, (for resetting the SDRAM's DLL) the corresponding bits of this field are ignored by the memory controller. Note that SDMODE[7] is mapped to MA[8].</p>

### 18.4.11 DDR SDRAM mode configuration 2 (DDR\_SDRAM\_MODE\_2)

### 18.4.11.1 Offset

Register	Offset
DDR_SDRAM_MODE_2	11Ch

### 18.4.11.2 Function

The DDR SDRAM mode 2 configuration register sets the values loaded into the DDR's extended mode 2 and 3 registers.

### 18.4.11.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ESDMODE3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.11.4 Fields

Field	Function
0-15 ESDMODE2	Extended SDRAM mode 2. Specifies the initial value loaded into the DDR SDRAM extended 2 mode register. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE2, which, in the big-endian convention shown here, corresponds to ESDMODE2[15]. The msb of the SDRAM extended mode 2 register value must be stored at ESDMODE2[0].
16-31 ESDMODE3	Extended SDRAM mode 3. Specifies the initial value loaded into the DDR SDRAM extended 3 mode register. The range of legal values of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of ESDMODE3, which, in the big-endian convention shown here, corresponds to ESDMODE3[15]. The msb of the SDRAM extended mode 3 register value must be stored at ESDMODE3[0].

## 18.4.12 DDR SDRAM mode control (DDR\_SDRAM\_MD\_CNTL)

### 18.4.12.1 Offset

Register	Offset
DDR_SDRAM_MD_CNT_L	120h

### 18.4.12.2 Function

The DDR SDRAM mode control register allows the user to carry out the following tasks:

Issue a mode register set command to a particular chip select

Issue an immediate refresh to a particular chip select

Issue an immediate precharge or precharge all command to a particular chip select

Force the CKE signals to a specific value

Note that MD\_EN, SET\_REF, and SET\_PRE are mutually exclusive; only one of these fields can be set at a time.

This table shows how DDR\_SDRAM\_MD\_CNTL fields should be set for each of the tasks described above.

**Table 18-6. Settings of DDR\_SDRAM\_MD\_CNTL Fields**

Field	Mode Register Set	Refresh	Precharge	Clock Enable Signals Control
MD_EN	1	0	0	-
SET_REF	0	1	0	-
SET_PRE	0	0	1	-
CS_SEL	Chooses chip select (CS)			-
MD_SEL	Select mode register.	-	Selects logical bank and bank group	-
MD_VALUE	Value written to mode register	-	Only bit 7 is significant.	-
CKE_CNTL	0	0	0	

### 18.4.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MD_EN	CS_SEL			MD_SEL				SET_RE							
W		L			L				F	E			Reserved	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									MD_VALUE							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.12.4 Fields

Field	Function
0 MD_EN	<p>Mode enable.</p> <p>Setting this bit specifies that valid data in MD_VALUE is ready to be written to DRAM as one of the following commands:</p> <ul style="list-style-type: none"> <li>• MODE REGISTER SET</li> <li>• EXTENDED MODE REGISTER SET</li> <li>• EXTENDED MODE REGISTER SET 2</li> <li>• EXTENDED MODE REGISTER SET 3</li> </ul> <p>The specific command to be executed is selected by setting MD_SEL. In addition, the chip select must be chosen by setting CS_SEL. MD_EN is set by software and cleared by hardware once the command has been issued.</p> <p>0b - Indicates that no mode register set command needs to be issued. 1b - Indicates that valid data contained in the register is ready to be issued as a mode register set command.</p>
1-3 CS_SEL	<p>Select chip select.</p> <p>Specifies the chip select that will be driven active due to any command forced by software in DDR_SDRAM_MD_CNTL.</p> <p>000b - Chip select 0 is active 001b - Chip select 1 is active 010b - Chip select 2 is active 011b - Chip select 3 is active 100b - Chip select 0 and chip select 1 are active 101b - Chip select 2 and chip select 3 are active 110b - Reserved 111b - Reserved</p>
4-7 MD_SEL	<p>Mode register select.</p> <p>MD_SEL specifies one of the following:</p> <ul style="list-style-type: none"> <li>• During a mode select command, selects the SDRAM mode register to be changed</li> </ul>

Table continues on the next page...

Field	Function
	<ul style="list-style-type: none"> <li>During a precharge command, selects the SDRAM logical bank to be precharged. A precharge all command ignores this field.</li> <li>During a refresh command, this field is ignored.</li> </ul> <p>In DDR3 mode, MD_SEL[1:3] contains the value that will be presented onto the memory bank address pins (MBA<math>n</math>) of the DDR controller. In DDR4 mode, MD_SEL[0:1] will represent the value on MBG[1:0], and MD_SEL[2:3] will represent the value on MBA[1:0].</p> <p>0000b - MR 0001b - EMR 0010b - EMR2 0011b - EMR3</p>
8 SET_REF	<p>Set refresh.</p> <p>Forces an immediate refresh to be issued to the chip select specified by DDR_SDRAM_MD_CNTL[CS_SEL]. This bit will be set by software and cleared by hardware once the command has been issued.</p> <p>0b - Indicates that no refresh command needs to be issued. 1b - Indicates that a refresh command is ready to be issued.</p>
9 SET_PRE	<p>Set precharge.</p> <p>Forces a precharge or precharge all to be issued to the chip select specified by DDR_SDRAM_MD_CNTL[CS_SEL]. This bit will be set by software and cleared by hardware once the command has been issued.</p> <p>0b - Indicates that no precharge all command needs to be issued. 1b - Indicates that a precharge all command is ready to be issued.</p>
10-11 CKE_CNTL	<p>Clock enable control.</p> <p>Allows software to globally clear or set the all CKE signals issued to DRAM. Once software has forced the value driven on CKE, that value will continue to be forced until software clears the CKE_CNTL bits. At that time, the DDR controller will continue to drive the CKE signals to the same value forced by software until another event causes the CKE signals to change (that is, self refresh entry/exit, power down entry/exit).</p> <p>00b - CKE signals are not forced by software. 01b - CKE signals are forced to a low value by software. 10b - CKE signals are forced to a high value by software. 11b - Reserved</p>
12 —	Reserved
13 —	Reserved
14-31 MD_VALUE	<p>Mode register value.</p> <p>This field, which specifies the value that will be presented on the memory address pins of the DDR controller during a mode register set command, is significant only when this register is used to issue a mode register set command or a precharge or precharge all command. Note that the 2 most significant bits of this register are implemented for future use, but they will not currently affect the MRS commands to the DRAM, as A[17:16] of the DRAMs MR registers are reserved.</p> <p>For a mode register set command, this field contains the data to be written to the selected mode register.</p> <p>For a precharge command, only bit five is significant:</p> <p>00000000000000000000b - Issue a precharge command; MD_SEL selects the logical bank to be precharged 00000000000000000001b - Issue a precharge all command; all logical banks are precharged</p>

## 18.4.13 DDR SDRAM interval configuration (DDR\_SDRAM\_INTERVAL)

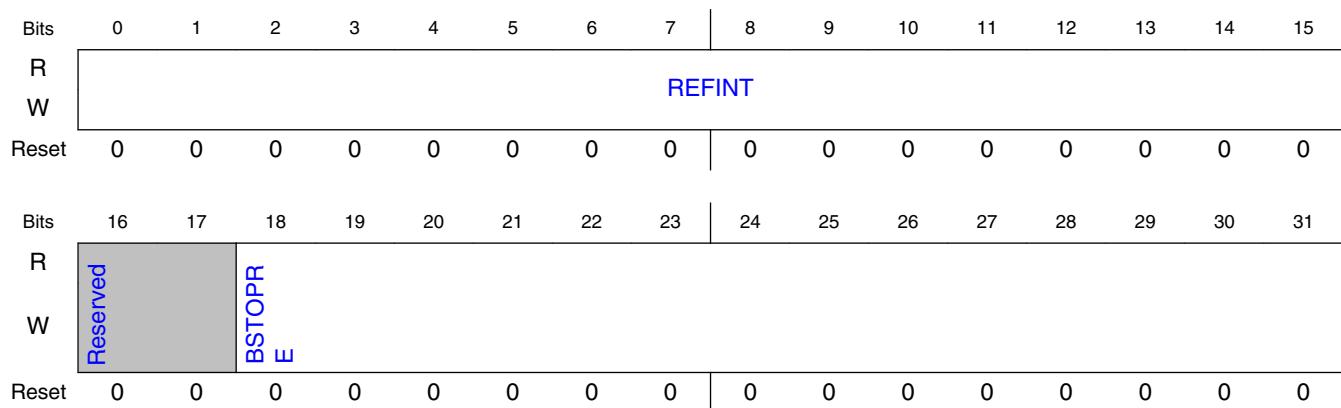
### 18.4.13.1 Offset

Register	Offset
DDR_SDRAM_INTERVAL	124h

### 18.4.13.2 Function

The DDR SDRAM interval configuration register sets the number of DRAM clock cycles between bank refreshes issued to the DDR SDRAMs. In addition, the number of DRAM cycles that a page is maintained after it is accessed is provided here.

### 18.4.13.3 Diagram



### 18.4.13.4 Fields

Field	Function
0-15	Refresh interval.
REFINT	

Table continues on the next page...

Field	Function
	Represents the number of memory bus clock cycles between refresh cycles. Depending on DDR_SDRAM_CFG_2[NUM_PR], some number of rows are refreshed in each DDR SDRAM physical bank during each refresh cycle. The value for REFINT depends on the specific SDRAMs used and the interface clock frequency. Refreshes will not be issued when the REFINT is set to all 0s. This field is concatenated with TIMING_CFG_4[EXT_REFINT] to obtain a 17-bit value for the total refresh interval.
16-17 —	Reserved
18-31	Precharge interval.
BSTOPRE	Sets the duration (in memory bus clocks) that a page is retained after a DDR SDRAM access. If BSTOPRE is zero, the DDR memory controller uses auto-precharge read and write commands rather than operating in page mode. This is called global auto-precharge mode.

## 18.4.14 DDR SDRAM data initialization (DDR\_DATA\_INIT)

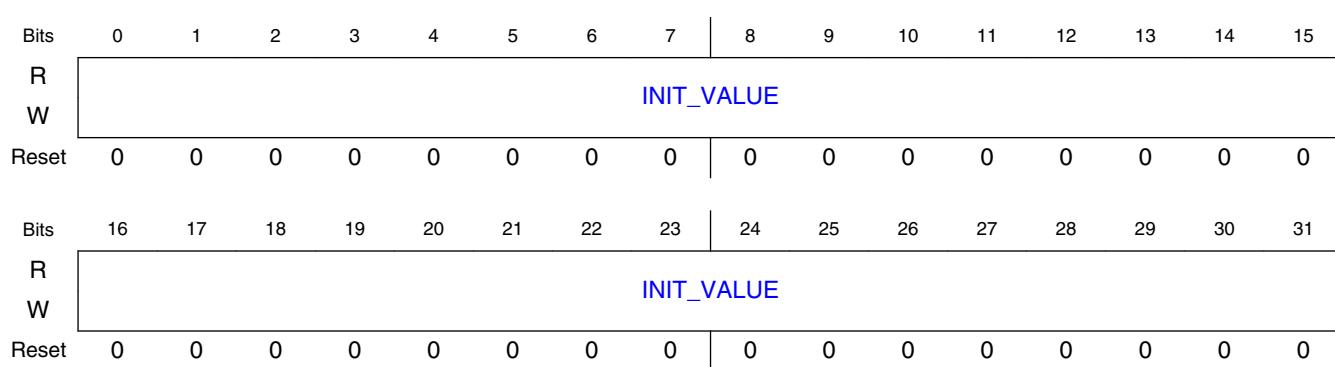
### 18.4.14.1 Offset

Register	Offset
DDR_DATA_INIT	128h

### 18.4.14.2 Function

The DDR SDRAM data initialization register provides the value that will be used to initialize memory if DDR\_SDRAM\_CFG2[D\_INIT] is set.

### 18.4.14.3 Diagram



### 18.4.14.4 Fields

Field	Function
0-31	Initialization value.
INIT_VALUE	Represents the value that DRAM will be initialized with if DDR_SDRAM_CFG2[D_INIT] is set.

## 18.4.15 DDR SDRAM clock control (DDR\_SDRAM\_CLK\_CNTL)

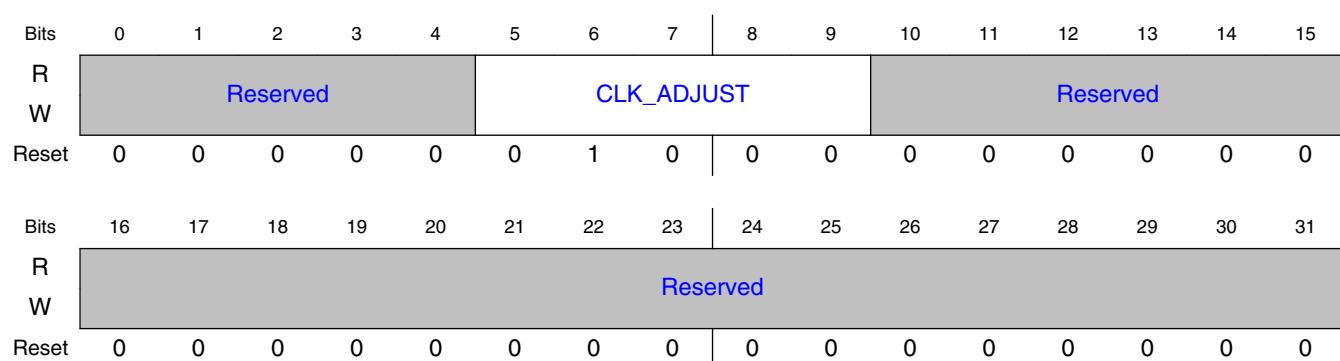
### 18.4.15.1 Offset

Register	Offset
DDR_SDRAM_CLK_CN TL	130h

### 18.4.15.2 Function

The DDR SDRAM clock control configuration register provides a 1/8-cycle clock adjustment.

### 18.4.15.3 Diagram



### 18.4.15.4 Fields

Field	Function
0-4 —	Reserved
5-9 CLK_ADJUST	Clock Adjust. 00000 Clock is launched aligned with address/command 00001 Clock is launched 1/16 applied cycle after address/command 00010 Clock is launched 1/8 applied cycle after address/command 00011 Clock is launched 3/16 applied cycle after address/command 00100 Clock is launched 1/4 applied cycle after address/command 00101 Clock is launched 5/16 applied cycle after address/command 00110 Clock is launched 3/8 applied cycle after address/command 00111 Clock is launched 7/16 applied cycle after address/command 01000 Clock is launched 1/2 applied cycle after address/command 01001 Clock is launched 9/16 applied cycle after address/command 01010 Clock is launched 5/8 applied cycle after address/command 01011 Clock is launched 11/16 applied cycle after address/command 01100 Clock is launched 3/4 applied cycle after address/command 01101 Clock is launched 13/16 applied cycle after address/command 01110 Clock is launched 7/8 applied cycle after address/command 01111 Clock is launched 15/16 applied cycle after address/command 10000 Clock is launched 1 applied cycle after address/command 10010-11110 Reserved
10-31 —	Reserved

### 18.4.16 DDR training initialization address (DDR\_INIT\_ADDR)

#### 18.4.16.1 Offset

Register	Offset
DDR_INIT_ADDR	148h

## 18.4.16.2 Function

The DDR SDRAM initialization address register provides the address that will be used for the data strobe to data skew adjustment and automatic CAS\_B to preamble calibration after POR.

When the default value is used (that is, address 0x0), all chip selects are considered for the training. If DDR\_INIT\_ADDR is set to any value other than the default value of address zero, then only the first chip select will be trained. When multiple chip selects are used and DQS/DQ skew is not common between chip selects/ranks, then the default address value of 0x0 is recommended to obtain the best timing margins.

After the skew adjustment, this address will contain bad ECC data. This is not important at POR, as all of memory should be subsequently initialized if ECC is enabled (either by software or through the use of DDR\_SDRAM\_CFG\_2[D\_INIT]).

If an HRESET\_B has been issued after the DRAM is in self-refresh mode, however, memory is not initialized, so this address should be written to using an 8- or 32-byte transaction to avoid possible ECC errors if this address could later be accessed.

## 18.4.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									INIT_ADDR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									INIT_ADDR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.16.4 Fields

Field	Function
0-31 INIT_ADDR	Initialization address. Represents the address that will be used for the data strobe to data skew adjustment and automatic CAS to preamble calibration at POR. This address will be written to during the initialization sequence.

## 18.4.17 DDR training initialization extended address (DDR\_INIT\_EXT\_ADDRESS)

### 18.4.17.1 Offset

Register	Offset
DDR_INIT_EXT_ADDRESS	14Ch

### 18.4.17.2 Function

The DDR SDRAM initialization extended address register provides the extended address that will be used for the data strobe to data skew adjustment and automatic CAS\_B to preamble calibration after POR.

### 18.4.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	UIA								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved				INIT_EXT_ADDR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.17.4 Fields

Field	Function
0 UIA	Use initialization address. 0b - Use the default address for training sequence as calculated by the controller. This will be the first valid address in each enabled chip select. 1b - Use the initialization address programmed in DDR_INIT_ADDR and DDR_INIT_EXT_ADDR.

*Table continues on the next page...*

## DDR register descriptions

Field	Function
1-23 —	Reserved
24-31 INIT_EXT_ADD_R	Initialization extended address. Represents the extended address that will be used for the data strobe to data skew adjustment and automatic CAS_B to preamble calibration at POR. This extended address will be written to during the initialization sequence.

## 18.4.18 DDR SDRAM timing configuration 4 (TIMING\_CFG\_4)

### 18.4.18.1 Offset

Register	Offset
TIMING_CFG_4	160h

### 18.4.18.2 Function

The DDR SDRAM timing configuration 4 register provides additional timing fields.

### 18.4.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RWT				WRT				RRT				WWT			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	EXT_RWT		Reserved	EXT_WRT	Reserved	EXT_RR_T	Reserved	EXT_WWT	Reserved			EXT_REFIN_T	Reserved	DLL_LOCK		
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.18.4 Fields

Field	Function
0-3 RWT	<p>Read-to-write turnaround for same chip select.</p> <p>Specifies how many cycles will be added between a read to write turnaround for transactions to the same chip select. If a value of 0000 is chosen, then the DDR controller will use the value used for transactions to different chip selects, as defined in TIMING_CFG_0[RWT]. This field can be used to improve performance when operating in burst-chop mode by forcing transactions to the same chip select to use extra cycles, while transaction to different chip selects can utilize the tri-state time on the DRAM interface. Regardless of the value that is set in this field, the value defined by TIMING_CFG_0[RWT] will also be met before issuing a write command.</p> <ul style="list-style-type: none"> <li>0000b - Default</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
4-7 WRT	<p>Write-to-read turnaround for same chip select.</p> <p>Specifies how many cycles will be added between a write to read turnaround for transactions to the same chip select. If a value of 0000 is chosen, then the DDR controller will use the value used for transactions to different chip selects, as defined in TIMING_CFG_0[WRT]. This field can be used to improve performance when operating in burst-chop mode by forcing transactions to the same chip select to use extra cycles, while transaction to different chip selects can utilize the tri-state time on the DRAM interface. Regardless of the value that is set in this field, the value defined by TIMING_CFG_0[WRT] will also be met before issuing a read command.</p> <ul style="list-style-type: none"> <li>0000b - Default</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
8-11 RRT	Read-to-read turnaround for same chip select.

*Table continues on the next page...*

## DDR register descriptions

Field	Function
	<p>Specifies how many cycles will be added between reads to the same chip select. If a value of 0000 is chosen, then 2 cycles will be required between read commands to the same chip select if 4-beat bursts are used (4 cycles will be required if 8-beat bursts are used). Note that fixed 4-beat bursts are not supported. However, this field may be used to add extra cycles when burst-chop mode is used, and the DDR controller must wait 4 cycles for read-to-read transactions to the same chip select.</p> <ul style="list-style-type: none"> <li>0000b - BL/2 clocks</li> <li>0001b - BL/2 + 1 clock</li> <li>0010b - BL/2 + 2 clocks</li> <li>0011b - BL/2 + 3 clocks</li> <li>0100b - BL/2 + 4 clocks</li> <li>0101b - BL/2 + 5 clocks</li> <li>0110b - BL/2 + 6 clocks</li> <li>0111b - BL/2 + 7 clocks</li> <li>1000b - BL/2 + 8 clocks</li> <li>1001b - BL/2 + 9 clocks</li> <li>1010b - BL/2 + 10 clocks</li> <li>1011b - BL/2 + 11 clocks</li> <li>1100b - BL/2 + 12 clocks</li> <li>1101b - BL/2 + 13 clocks</li> <li>1110b - BL/2 + 14 clocks</li> <li>1111b - BL/2 + 15 clocks</li> </ul>
12-15 WWT	<p>Write-to-write turnaround for same chip select.</p> <p>Specifies how many cycles will be added between writes to the same chip select. If a value of 0000 is chosen, then 2 cycles will be required between write commands to the same chip select if 4-beat bursts are used (4 cycles will be required if 8-beat bursts are used). Note that fixed 4-beat bursts are not supported. However, this field may be used to add extra cycles when burst-chop mode is used, and the DDR controller must wait 4 cycles for write-to-write transactions to the same chip select.</p> <ul style="list-style-type: none"> <li>0000b - BL/2 clocks</li> <li>0001b - BL/2 + 1 clock</li> <li>0010b - BL/2 + 2 clocks</li> <li>0011b - BL/2 + 3 clocks</li> <li>0100b - BL/2 + 4 clocks</li> <li>0101b - BL/2 + 5 clocks</li> <li>0110b - BL/2 + 6 clocks</li> <li>0111b - BL/2 + 7 clocks</li> <li>1000b - BL/2 + 8 clocks</li> <li>1001b - BL/2 + 9 clocks</li> <li>1010b - BL/2 + 10 clocks</li> <li>1011b - BL/2 + 11 clocks</li> <li>1100b - BL/2 + 12 clocks</li> <li>1101b - BL/2 + 13 clocks</li> <li>1110b - BL/2 + 14 clocks</li> <li>1111b - BL/2 + 15 clocks</li> </ul>
16-17 EXT_RWT	<p>Extended read-to-write turnaround (tRTW).</p> <p>Specifies how many extra cycles will be added between a read to write turnaround. If 0 clocks is chosen, then the DDR controller will use a fixed number based on the CAS latency and write latency. Choosing a value other than 0 adds extra cycles past this default calculation. As a default the DDR controller will determine the read-to-write turnaround as <math>CL - WL + BL/2 + 2</math>. In this equation, CL is the CAS latency rounded up to the next integer, WL is the programmed write latency, and BL is the burst length. This field is concatenated with TIMING_CFG_0[RWT] to obtain a 4-bit value for the total read-to-write turnaround</p> <ul style="list-style-type: none"> <li>00b - 0 clocks</li> <li>01b - 1 clock</li> <li>10b - 2 clocks</li> <li>11b - 3 clocks</li> </ul>

Table continues on the next page...

Field	Function
18 —	Reserved
19 EXT_WRT	<p>Extended write-to-read turnaround.</p> <p>Specifies how many extra cycles will be added between a write to read turnaround. If 0 clocks is chosen, then the DDR controller will use a fixed number based on the, read latency, and write latency. Choosing a value other than 0 adds extra cycles past this default calculation. As a default, the DDR controller will determine the write-to-read turnaround as <math>WL - CL + BL/2 + 1</math>. In this equation, CL is the CAS latency rounded down to the next integer, WL is the programmed write latency, and BL is the burst length. This field is concatenated with TIMING_CFG_0[WRT] to obtain a 3-bit value for the total write-to-read turnaround</p> <p>0b - 0 clocks 1b - 4 clocks</p>
20 —	Reserved
21 EXT_RRT	<p>Extended read-to-read turnaround.</p> <p>Specifies how many extra cycles will be added between reads to different chip selects. As a default, 3 cycles will be required between read commands to different chip selects. Extra cycles may be added with this field. Note: If 8-beat bursts are enabled, then 5 cycles will be the default. This field is concatenated with TIMING_CFG_0[RRT] to obtain a 3-bit value for the total read-to-read turnaround</p> <p>0b - 0 clocks 1b - 4 clocks</p>
22 —	Reserved
23 EXT_WWT	<p>Extended write-to-write turnaround.</p> <p>Specifies how many extra cycles will be added between writes to different chip selects. As a default, 2 cycles will be required between write commands to different chip selects. Extra cycles may be added with this field. Note: If 8-beat bursts are enabled, then 4 cycles will be the default. This field is concatenated with TIMING_CFG_0[WWT] to obtain a 3-bit value for the total write-to-write turnaround</p> <p>0b - 0 clocks 1b - 4 clocks</p>
24-26 —	Reserved
27 EXT_REFINT	<p>Extended Refresh Interval.</p> <p>Represents the number of memory bus clock cycles between refresh cycles. Depending on DDR_SDRAM_CFG_2[NUM_PR], some number of rows are refreshed in each DDR SDRAM physical bank during each refresh cycle. The value for REFINT depends on the specific SDRAMs used and the interface clock frequency. Refreshes will not be issued when the REFINT is set to all 0s. This field is concatenated with DDR_SDRAM_INTERVAL[REFINT] to obtain a 17-bit value for the total refresh interval.</p> <p>0b - 0 clocks 1b - 65,536 clocks</p>
28-29 —	Reserved
30-31 DLL_LOCK	DDR SDRAM DLL Lock Time.

## DDR register descriptions

Field	Function
	<p>This provides the number of cycles that it will take for the DRAMs DLL to lock at POR and after exiting self refresh. The controller will wait the specified number of cycles before issuing any commands after exiting POR or self refresh.</p> <p>00b - 200 clocks 01b - 512 clocks 10b - 1024 clocks 11b - Reserved</p>

## 18.4.19 DDR SDRAM timing configuration 5 (TIMING\_CFG\_5)

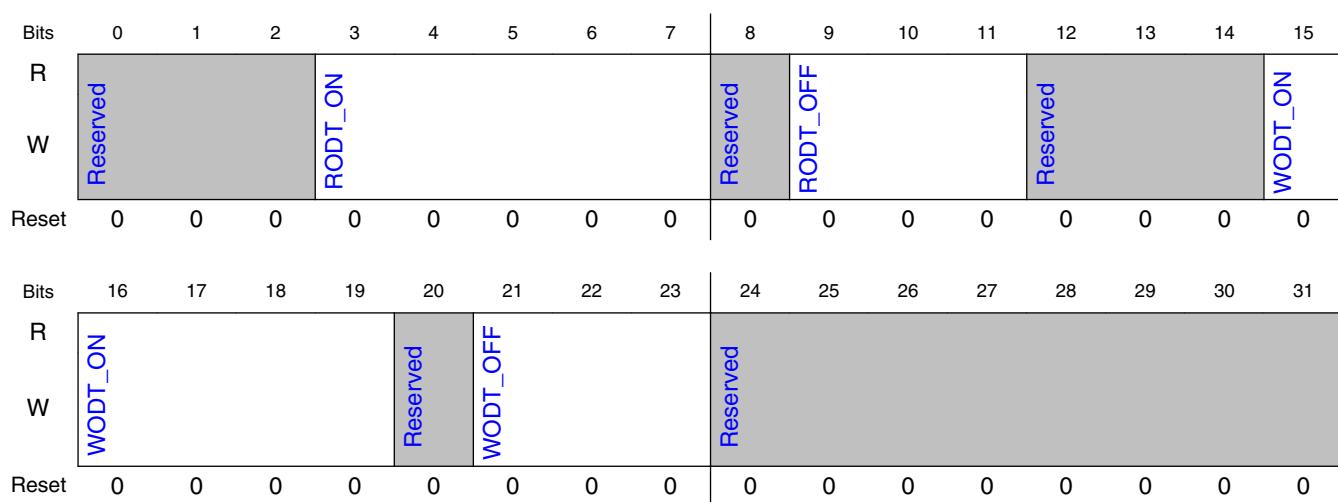
### 18.4.19.1 Offset

Register	Offset
TIMING_CFG_5	164h

### 18.4.19.2 Function

The DDR SDRAM timing configuration 5 register provides additional timing fields.

### 18.4.19.3 Diagram



## 18.4.19.4 Fields

Field	Function
0-2 —	Reserved
3-7 RODT_ON	<p>Read to ODT on.</p> <p>Specifies the number of cycles that will pass from when a read command is placed on the DRAM bus until the assertion of the relevant ODT signal(s). The default case (00000) provides a decode of [CASLAT - WR_LAT] to provide the expected default. If 2T timing is used, an extra cycle will automatically be added to the value selected in this field.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>00000b - CASLAT - WR_LAT</li> <li>00001b - 0 clocks</li> <li>00010b - 1 clock</li> <li>00011b - 2 clocks</li> <li>01100b - 11 clocks</li> </ul>
8 —	Reserved
9-11 RODT_OFF	<p>Read to ODT off.</p> <p>Specifies the number of cycles that the relevant ODT signal(s) will remain asserted for each read transaction. The default case (000) will leave the ODT signal(s) asserted for 4 DRAM cycles.</p> <ul style="list-style-type: none"> <li>000b - 4 clocks</li> <li>001b - 1 clock</li> <li>010b - 2 clocks</li> <li>011b - 3 clocks</li> <li>100b - 4 clocks</li> <li>101b - 5 clocks</li> <li>110b - 6 clocks</li> <li>111b - 7 clocks</li> </ul>
12-14 —	Reserved
15-19 WODT_ON	<p>Write to ODT on.</p> <p>Specifies the number of cycles that will pass from when a write command is placed on the DRAM bus until the assertion of the relevant ODT signal(s). The default case (00000) provides a decode of 0 cycles to provide the expected default. If 2T timing is used, an extra cycle will automatically be added to the value selected in this field.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>00000b - 0 clocks</li> <li>00001b - 0 clocks</li> <li>00010b - 1 clock</li> <li>00011b - 2 clocks</li> <li>00110b - 5 clocks</li> </ul>
20 —	Reserved
21-23 WODT_OFF	<p>Write to ODT off.</p> <p>Specifies the number of cycles that the relevant ODT signal(s) will remain asserted for each write transaction. The default case (000) will leave the ODT signal(s) asserted for 4 DRAM cycles.</p>

*Table continues on the next page...*

Field	Function
	<b>NOTE:</b> Value of 4 clock cycles must be selected for this field. 000b - 4 clocks 001b - 1 clock 010b - 2 clocks 011b - 3 clocks 100b - 4 clocks 101b - 5 clocks 110b - 6 clocks 111b - 7 clocks
24-31 —	Reserved

## 18.4.20 DDR SDRAM timing configuration 6 (TIMING\_CFG\_6)

### 18.4.20.1 Offset

Register	Offset
TIMING_CFG_6	168h

### 18.4.20.2 Function

The DDR SDRAM timing configuration 6 register provides additional timing fields.

If setting TIMING\_CFG\_6[HS\_CASLAT] or TIMING\_CFG\_6[HS\_WRLAT] to a non-zero value, then the additive latency must be programmed to 0 clocks.

If setting TIMING\_CFG\_6[HS\_WRLAT] to a non-zero value, then TIMING\_CFG\_5[WODT\_ON] should be programmed to 0 clocks.

### 18.4.20.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			HS_CASLAT					HS_WRLAT					Reserved		HS_WRRE
W																C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	HS_WRRE				Reserved	Reserved					Reserved	Reserved				
W	C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.20.4 Fields

Field	Function
0-2 —	Reserved
3-7 HS_CASLAT	<p>Half-Speed CAS Latency.</p> <p>MCAS_B latency from READ command while DDR controller is operating at half frequency. Number of clock cycles between registration of a READ command by the SDRAM and the availability of the first output data. If a READ command is registered at clock edge <math>n</math> and the latency is <math>m</math> clocks, data is available nominally coincident with clock edge <math>n + m</math>. If a non-zero field of this register is used, then the additive latency (TIMING_CFG_2[ADD_LAT] must be programmed to 0 clocks).</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>00000b - Use full speed value</li> <li>00101b - 5 clocks</li> <li>00110b - 6 clocks</li> <li>00111b - 7 clocks</li> <li>01000b - 8 clocks</li> <li>01001b - 9 clocks</li> <li>01010b - 10 clocks</li> <li>01011b - 11 clocks</li> <li>01100b - 12 clocks</li> <li>01101b - 13 clocks</li> <li>01110b - 14 clocks</li> <li>01111b - 15 clocks</li> <li>10000b - 16 clocks</li> <li>10001b - 17 clocks</li> <li>10010b - 18 clocks</li> <li>10011b - 19 clocks</li> <li>10100b - 20 clocks</li> </ul>
8-12	Half-Speed Write Latency.

Table continues on the next page...

## DDR register descriptions

Field	Function
HS_WRLAT	<p>Write latency while DDR controller is operating at half frequency. Note that the total write latency is equal to WR_LAT + ADD_LAT. Note that the total write latency must be at least 6 cycles if using unbuffered DIMMs in 1T timing mode. Note that this is not a straight decode, as bit 13 of this register sets the msb of the WR_LAT field. If using a non-zero value for this field, then TIMING_CFG_5[WODT_ON] must be set to 0 clocks to ensure ODT is driven correctly for writes.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>00000b - Use full speed value</li> <li>00001b - 16 clocks</li> <li>00011b - 17 clocks</li> <li>00101b - 18 clocks</li> <li>01010b - 5 clocks</li> <li>01100b - 6 clocks</li> <li>01110b - 7 clocks</li> </ul>
13-14 —	Reserved
15-19	Half-Speed Write Recovery.
HS_WRREC	<p>Last data to precharge minimum interval (<math>t_{WR}</math>) while DDR controller is operating at half frequency. Determines the number of clock cycles from the last data associated with a write command until a precharge command is allowed. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to (<math>t_{WR} + 2</math> cycles).</p> <ul style="list-style-type: none"> <li>00000b - Use full speed value.</li> <li>00001b - 1 clock</li> <li>00010b - 2 clocks</li> <li>00011b - 3 clocks</li> <li>00100b - 4 clocks</li> <li>00101b - 5 clocks</li> <li>00110b - 6 clocks</li> <li>00111b - 7 clocks</li> <li>01000b - 8 clocks</li> <li>01001b - 9 clocks</li> <li>01010b - 10 clocks</li> <li>01011b - 11 clocks</li> <li>01100b - 12 clocks</li> <li>01101b - 13 clocks</li> <li>01110b - 14 clocks</li> <li>01111b - 15 clocks</li> <li>10000b - 16 clocks</li> <li>10001b - 17 clocks</li> <li>10010b - 18 clocks</li> <li>10011b - 19 clocks</li> <li>10100b - 20 clocks</li> <li>10101b - 21 clocks</li> <li>10110b - 22 clocks</li> <li>10111b - 23 clocks</li> <li>11000b - 24 clocks</li> <li>11001b - 25 clocks</li> <li>11010b - 26 clocks</li> <li>11011b - 27 clocks</li> <li>11100b - 28 clocks</li> <li>11101b - 29 clocks</li> <li>11110b - 30 clocks</li> <li>11111b - 31 clocks</li> </ul>
20	Reserved

Table continues on the next page...

Field	Function
—	
21-25	Reserved
—	
26	Reserved
—	
27-31	Reserved
—	

## 18.4.21 DDR SDRAM timing configuration 7 (TIMING\_CFG\_7)

### 18.4.21.1 Offset

Register	Offset
TIMING_CFG_7	16Ch

### 18.4.21.2 Function

The DDR SDRAM timing configuration 7 register provides additional timing fields required.

### 18.4.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W			CKE_RS T		CKSR E				CKSR X				PAR_LAT			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved					CS_TO_CMD				Reserved			
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.21.4 Fields

Field	Function
0-1 —	Reserved
2-3 CKE_RST	<p>CKE Reset Time.</p> <p>CKE reset time (<math>t_{XPR}</math>). Specifies the number of cycles the DDR controller must wait after asserting CKE after RESET until the first MRS command. This field is also used to determine <math>t_{XS}</math> when exiting self refresh. Therefore, this should be programmed to the maximum of (<math>t_{XPR}</math>, <math>t_{XS}</math>) if they are different for a particular memory.</p> <ul style="list-style-type: none"> <li>00b - 200 clocks</li> <li>01b - 256 clocks</li> <li>10b - 512 clocks</li> <li>11b - 1024 clocks</li> </ul>
4-7 CKSRE	<p>Clock After Self Refresh Entry.</p> <p>Valid clock after Self Refresh entry (<math>t_{CKSRE}</math>). Specifies the number of cycles the DDR controller must drive valid MCK/MCK_B after entering self refresh before the clocks are allowed to stop.</p> <p><b>TIMING_CFG_7[CS_TO_CMD]</b> should also be added to <math>t_{CKSRE}</math> to obtain the final value for this field.</p> <ul style="list-style-type: none"> <li>0000b - 15 clocks</li> <li>0001b - 6 clocks</li> <li>0010b - 7 clocks</li> <li>0011b - 8 clocks</li> <li>0100b - 9 clocks</li> <li>0101b - 10 clocks</li> <li>0110b - 11 clocks</li> <li>0111b - 12 clocks</li> <li>1000b - 13 clocks</li> <li>1001b - 14 clocks</li> <li>1010b - 15 clocks</li> <li>1011b - 16 clocks</li> <li>1100b - 17 clocks</li> <li>1101b - 18 clocks</li> <li>1110b - 19 clocks</li> <li>1111b - 32 clocks</li> </ul>
8-11 CKSRX	<p>Clock After Self Refresh Exit.</p> <p>Valid clock after Self Refresh exit (<math>t_{CKSRX}</math>). Specifies the number of cycles the DDR controller must drive valid MCK/MCK_B after exiting self refresh before the clocks are allowed to stop.</p> <ul style="list-style-type: none"> <li>0000b - 15 clocks</li> <li>0001b - 6 clocks</li> <li>0010b - 7 clocks</li> <li>0011b - 8 clocks</li> <li>0100b - 9 clocks</li> <li>0101b - 10 clocks</li> <li>0110b - 11 clocks</li> <li>0111b - 12 clocks</li> <li>1000b - 13 clocks</li> <li>1001b - 14 clocks</li> <li>1010b - 15 clocks</li> <li>1011b - 16 clocks</li> <li>1100b - 17 clocks</li> <li>1101b - 18 clocks</li> <li>1110b - 19 clocks</li> </ul>

Table continues on the next page...

Field	Function
	1111b - 27 clocks
12-15 PAR_LAT	<p>Parity latency.</p> <p>Specifies the number of cycles to be used for the parity latency for DDR4 memories. For DDR3 memory types, this should be disabled.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>0000b - Disabled</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> </ul>
16-23 —	Reserved
24-27 CS_TO_CMD	<p>Chip select to command latency.</p> <p>Specifies the number of cycles from a chip select until the command is launched. This should only be used with DDR4. The DDR controller will automatically apply this latency after it has written the MR4 register during DDR4 SDRAM initialization. It will also be enabled when the controller is enabled if DDR_SDRAM_CFG[B1] is set. However, if software is going to use the DDR_SDRAM_MD_CNTL register to initialize the DRAM's MR registers instead of allowing for automatic calibration, then this field cannot be set until software has properly programmed the DDR4 MR4 register to enable CAL mode.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>0000b - Disabled</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> </ul>
28-31 —	Reserved

## 18.4.22 DDR ZQ calibration control (DDR\_ZQ\_CNTL)

### 18.4.22.1 Offset

Register	Offset
DDR_ZQ_CNTL	170h

### 18.4.22.2 Function

The DDR ZQ Calibration Control register provides the enable and controls required for ZQ calibration.

There is a limitation for various DRAM timing parameters when ZQ calibration is used. The factors involved in this limitation are DDR\_ZQ\_CNTL[ZQOPER], DDR\_ZQ\_CNTL[ZQCS], TIMING\_CFG\_1[PRETOACT], TIMING\_CFG\_1[REFREC], DDR\_SDRAM\_INTERVAL[REFINT], and the number of chip selects enabled. If the following condition is true:

$$[((\text{DDR\_ZQ\_CNTL}[ZQOPER] + \text{DDR\_ZQ\_CNTL}[ZQCS]) * (\# \text{ enabled chip selects})) + \text{TIMING\_CFG\_1}[PRETOACT] + \text{TIMING\_CFG\_1}[REFREC] + 2t_{CK}] > (\text{DDR\_SDRAM\_INTERVAL}[REFINT]),$$

then it is possible that one refresh will be skipped when the controller is exiting self refresh. If this is an issue, then posted refreshes could be used to extend the refresh interval. Another alternative is to use the DDR\_SDRAM\_MD\_CNTL register to force an extra refresh to each chip select after exiting self refresh mode. However, timing parameters for most devices/frequencies will not allow for a refresh to be missed.

### 18.4.22.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ZQ_EN	Reserved			ZQINIT				Reserved				ZQOPER			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				ZQCS				Reserved				ZQCS_INT			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.22.4 Fields

Field	Function
0	ZQ Calibration Enable.

*Table continues on the next page...*

Field	Function
ZQ_EN	This bit determines if ZQ calibrating will be used. 0b - ZQ Calibration will not be used. 1b - ZQ Calibration will be used. A ZQCL command will be issued by the DDR controller after POR and anytime the DDR controller is exiting self refresh. A ZQCS command will be issued every 32 refresh sequences to account for VT variations.
1-3 —	Reserved
4-7 ZQINIT	ZQ Calibration Initialization Time. POR ZQ Calibration Time ( $t_{ZQinit}$ ). Determines the number of cycles that must be allowed for DRAM ZQ calibration at POR. Each chip select will be calibrated separately, and this time must elapse after the ZQCL command is issued for each chip select before a separate command may be issued. Patterns not shown are reserved. 0111b - 128 clocks 1000b - 256 clocks 1001b - 512 clocks 1010b - 1024 clocks
8-11 —	Reserved
12-15 ZQOPER	ZQ Calibration Operation Time. Normal Operation Full Calibration Time ( $t_{ZQoper}$ ). Determines the number of cycles that must be allowed for DRAM ZQ calibration when exiting self refresh. Each chip select will be calibrated separately, and this time must elapse after the ZQCL command is issued for each chip select before a separate command may be issued. Patterns not shown are reserved. 0111b - 128 clocks 1000b - 256 clocks 1001b - 512 clocks 1010b - 1024 clocks
16-19 —	Reserved
20-23 ZQCS	ZQ Calibration Short Time. Normal Operation Short Calibration Time ( $t_{ZQCS}$ ). Determines the number of cycles that must be allowed for DRAM ZQ calibration during dynamic calibration which is issued every ZQCS_INT refresh sequences. Each chip select will be calibrated separately, and this time must elapse after the ZQCS command is issued for each chip select before a separate command may be issued. Patterns not shown are reserved. 0000b - 1 clock 0001b - 2 clocks 0010b - 4 clocks 0011b - 8 clocks 0100b - 16 clocks 0101b - 32 clocks 0110b - 64 clocks 0111b - 128 clocks 1000b - 256 clocks 1001b - 512 clocks
24-27 —	Reserved

*Table continues on the next page...*

## DDR register descriptions

Field	Function
28-31 ZQCS_INT	ZQCS Interval.  Determines the number of refresh sequences that will pass between each ZQCS calibration.  0000b - 32 refresh sequences 0001b - 64 refresh sequences 0010b - 128 refresh sequences 0011b - 256 refresh sequences 0100b - 512 refresh sequences 0101b - 1024 refresh sequences 0110b - 2048 refresh sequences 0111b - 4096 refresh sequences 1000b - 8192 refresh sequences 1001b - 16384 refresh sequences 1010b - 32768 refresh sequences 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - ZQCS calibration disabled

## 18.4.23 DDR write leveling control (DDR\_WRLVL\_CNTL)

### 18.4.23.1 Offset

Register	Offset
DDR_WRLVL_CNTL	174h

### 18.4.23.2 Function

The DDR Write Leveling Control register provides controls for write leveling.

### 18.4.23.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<code>WRLVL_EN</code>	<code>WRLVL_DONE</code>	Reserved			<code>WRLVL_MRД</code>			Reserved	<code>WRLVL_ODTEN</code>			Reserved	<code>WRLVL_DQSEN</code>		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	<code>WRLVL_SMPL</code>				Reserved	<code>WRLVL_WLR</code>			Reserved			<code>WRLVL_START</code>				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.23.4 Fields

Field	Function
0 WRLVL_EN	Write Leveling Enable. This bit determines if write leveling will be used. If this bit is set, then the DDR controller will perform write leveling immediately after initializing the DRAM. 0b - Write leveling will not be used 1b - Write leveling will be used
1 WRLVL_DONE	Write Leveling Done. This bit will be set by hardware once write leveling has been completed. This is a read-only bit, and it will only clear after a reset to the part has been issued. 0b - Write leveling has not completed 1b - Write leveling has completed
2-4 —	Reserved
5-7 WRLVL_MRД	Write Leveling MRД. First DQS pulse rising edge after margining mode is programmed ( $t_{WL\_MRД}$ ). Determines how many cycles to wait after margining mode has been programmed before the first DQS pulse may be issued. This field is only relevant when DDR_WRLVL_CNTL[WRLVL_EN] is set. 000b - 1 clock 001b - 2 clocks 010b - 4 clocks 011b - 8 clocks 100b - 16 clocks

Table continues on the next page...

## DDR register descriptions

Field	Function
	101b - 32 clocks 110b - 64 clocks 111b - 128 clocks
8 —	Reserved
9-11 WRLVL_ODTE N	Write Leveling ODT Enable. ODT delay after margining mode is programmed ( $t_{WL\_ODTEN}$ ). Determines how many cycles to wait after margining mode has been programmed.until ODT may be asserted.This field is only relevant when DDR_WRLVL_CNTL[WRLVL_EN] is set. 000b - 1 clock 001b - 2 clocks 010b - 4 clocks 011b - 8 clocks 100b - 16 clocks 101b - 32 clocks 110b - 64 clocks 111b - 128 clocks
12 —	Reserved
13-15 WRLVL_DQSE N	Write Leveling DQS Enable. DQS/DQS_B delay after margining mode is programmed ( $t_{WL\_DQSEN}$ ). Determines how many cycles to wait after margining mode has been programmed.until DQS may be actively driven. This field is only relevant when DDR_WRLVL_CNTL[WRLVL_EN] is set. 000b - 1 clock 001b - 2 clocks 010b - 4 clocks 011b - 8 clocks 100b - 16 clocks 101b - 32 clocks 110b - 64 clocks 111b - 128 clocks
16-19 WRLVL_SMPL	Write leveling sample time. Determines the number of cycles that must pass before the data signals are sampled after a DQS pulse during margining mode. This field should be programmed at least 6 cycles higher than $t_{WL0}$ to allow enough time for propagation delay and sampling of the prime data bits. This field is only relevant when DDR_WRLVL_CNTL[WRLVL_EN] is set. 0000b - 32 clocks 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks

Table continues on the next page...

Field	Function
	1111b - 15 clocks
20 —	Reserved
21-23 WRLVL_WLR	<p>Write leveling repetition time.</p> <p>Determines the number of cycles that must pass between DQS pulses during write leveling. This field is only relevant when DDR_WRLVL_CNTL[WRLVL_EN] is set.</p> <ul style="list-style-type: none"> <li>000b - 1 clock</li> <li>001b - 2 clocks</li> <li>010b - 4 clocks</li> <li>011b - 8 clocks</li> <li>100b - 16 clocks</li> <li>101b - 32 clocks</li> <li>110b - 64 clocks</li> <li>111b - 128 clocks</li> </ul>
24-26 —	Reserved
27-31 WRLVL_START	<p>Write Leveling Start for DQS[0].</p> <p>Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled.</p> <ul style="list-style-type: none"> <li>00000b - 0 clock delay</li> <li>00001b - 1/8 clock delay</li> <li>00010b - 1/4 clock delay</li> <li>00011b - 3/8 clock delay</li> <li>00100b - 1/2 clock delay</li> <li>00101b - 5/8 clock delay</li> <li>00110b - 3/4 clock delay</li> <li>00111b - 7/8 c'lock delay</li> <li>01000b - 1 clock delay</li> <li>01001b - 9/8 clock delay</li> <li>01010b - 5/4 clock delay</li> <li>01011b - 11/8 clock delay</li> <li>01100b - 3/2 clock delay</li> <li>01101b - 13/8 clock delay</li> <li>01110b - 7/4 clock delay</li> <li>01111b - 15/8 clock delay</li> <li>10000b - 2 clock delay</li> <li>10001b - 17/8 clock delay</li> <li>10010b - 9/4 clock delay</li> <li>10011b - 19/8 clock delay</li> <li>10100b - 5/2 clock delay</li> <li>10101b - 21/8 clock delay</li> <li>10110b - 16/4 clock delay</li> <li>10111b - 23/8 clock delay</li> <li>11000b - 3 clock delay</li> <li>11001b - 25/8 clock delay</li> <li>11010b - 13/4 clock delay</li> <li>11011b - 27/8 clock delay</li> <li>11100b - 7/2 clock delay</li> <li>11101b - 29/8 clock delay</li> <li>11110b - 15/4 clock delay</li> <li>11111b - 31/8 clock delay</li> </ul>

## 18.4.24 DDR Self Refresh Counter (DDR\_SR\_CNTR)

### 18.4.24.1 Offset

Register	Offset
DDR_SR_CNTR	17Ch

### 18.4.24.2 Function

The DDR Self Refresh Counter register can be programmed to force the DDR controller to enter self refresh after a predefined period of idle time.

### 18.4.24.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.24.4 Fields

Field	Function
0-11 —	Reserved
12-15 SR_IT	Self Refresh Idle Threshold. Defines the number of DRAM cycles that must pass while the DDR controller is idle before it will enter self refresh. Anytime a transaction is issued to the DDR controller, it will reset its internal counter. When a new transaction is received by the DDR controller, it will exit self refresh and reset its internal counter. If

*Table continues on the next page...*

Field	Function
	<p>this field is zero, then the described power savings feature will be disabled. In addition, if a non-zero value is programmed into this field, then the DDR controller will exit self refresh anytime a transaction is issued to the DDR controller, regardless of the reason self refresh was initially entered.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>0000b - Automatic self refresh entry disabled</li> <li>0001b - <math>2^{10}</math> DRAM clocks</li> <li>0010b - <math>2^{12}</math> DRAM clocks</li> <li>0011b - <math>2^{14}</math> DRAM clocks</li> <li>0100b - <math>2^{16}</math> DRAM clocks</li> <li>0101b - <math>2^{18}</math> DRAM clocks</li> <li>0110b - <math>2^{20}</math> DRAM clocks</li> <li>0111b - <math>2^{22}</math> DRAM clocks</li> <li>1000b - <math>2^{24}</math> DRAM clocks</li> <li>1001b - <math>2^{26}</math> DRAM clocks</li> <li>1010b - <math>2^{28}</math> DRAM clocks</li> <li>1011b - <math>2^{30}</math> DRAM clocks</li> </ul>
16-31 —	Reserved

## 18.4.25 DDR Register Control Words 1 (DDR\_SDRAM\_RCW\_1)

### 18.4.25.1 Offset

Register	Offset
DDR_SDRAM_RCW_1	180h

### 18.4.25.2 Function

The DDR Register Control Word 1 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 4-bit field represents the value that will be placed on MBA[1], MBA[0], MA[4], and MA[3] in DDR3 mode during register control word writes. Each 4-bit field represents the value that will be placed on MA[3:0] in DDR4 mode during register control word writes.

### 18.4.25.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW0				RCW1				RCW2				RCW3			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW4				RCW5				RCW6				RCW7			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.25.4 Fields

Field	Function
0-3	Register Control Word 0.
RCW0	Represents the value that will be used during writes to register control word 0.
4-7	Register Control Word 1.
RCW1	Represents the value that will be used during writes to register control word 1.
8-11	Register Control Word 2.
RCW2	Represents the value that will be used during writes to register control word 2.
12-15	Register Control Word 3.
RCW3	Represents the value that will be used during writes to register control word 3.
16-19	Register Control Word 4.
RCW4	Represents the value that will be used during writes to register control word 4.
20-23	Register Control Word 5.
RCW5	Represents the value that will be used during writes to register control word 5.
24-27	Register Control Word 6.
RCW6	Represents the value that will be used during writes to register control word 6.
28-31	Register Control Word 7.
RCW7	Represents the value that will be used during writes to register control word 7.

## 18.4.26 DDR Register Control Words 2 (DDR\_SDRAM\_RCW\_2)

### 18.4.26.1 Offset

Register	Offset
DDR_SDRAM_RCW_2	184h

### 18.4.26.2 Function

The DDR Register Control Word 2 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 4-bit field represents the value that will be placed on MBA[1], MBA[0], MA[4], and MA[3] in DDR3 mode during register control word writes. Each 4-bit field represents the value that will be placed on MA[3:0] in DDR4 mode during register control word writes.

t

### 18.4.26.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW8				RCW9				RCW10				RCW11			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW12				RCW13				RCW14				RCW15			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.26.4 Fields

Field	Function
0-3	Register Control Word 8.
RCW8	Represents the value that will be used during writes to register control word 8.
4-7	Register Control Word 9.
RCW9	Represents the value that will be used during writes to register control word 9.
8-11	Register Control Word 10.
RCW10	Represents the value that will be used during writes to register control word 10.

Table continues on the next page...

## DDR register descriptions

Field	Function
12-15 RCW11	Register Control Word 11. Represents the value that will be used during writes to register control word 11.
16-19 RCW12	Register Control Word 12. Represents the value that will be used during writes to register control word 12.
20-23 RCW13	Register Control Word 13. Represents the value that will be used during writes to register control word 13.
24-27 RCW14	Register Control Word 14. Represents the value that will be used during writes to register control word 14.
28-31 RCW15	Register Control Word 15. Represents the value that will be used during writes to register control word 15.

## 18.4.27 DDR write leveling control 2 (DDR\_WRLVL\_CNTL\_2)

### 18.4.27.1 Offset

Register	Offset
DDR_WRLVL_CNTL_2	190h

### 18.4.27.2 Function

The DDR Write Leveling Control 2 register provides controls for write leveling. This register specifically defines the starting points for the individual data strobes.

Note: For each field WRLVL\_START\_n, a setting of 0000 is not recommended; it disables the per-byte write level start time selection. It is recommended to select proper individual delay for each byte lane based on board skews.

### 18.4.27.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			WRLVL_START_1				Reserved			WRLVL_START_2					
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			WRLVL_START_3				Reserved			WRLVL_START_4					
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.27.4 Fields

Field	Function
0-2 —	Reserved
3-7 WRLVL_START_1	<p>Write leveling start time for DQS[1].</p> <p>Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled.</p> <p>00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START]      00001b - 1/8 clock delay      00010b - 1/4 clock delay      00011b - 3/8 clock delay      00100b - 1/2 clock delay      00101b - 5/8 clock delay      00110b - 3/4 clock delay      00111b - 7/8 c'lock delay      01000b - 1 clock delay      01001b - 9/8 clock delay      01010b - 5/4 clock delay      01011b - 11/8 clock delay      01100b - 3/2 clock delay      01101b - 13/8 clock delay      01110b - 7/4 clock delay      01111b - 15/8 clock delay      10000b - 2 clock delay      10001b - 17/8 clock delay      10010b - 9/4 clock delay      10011b - 19/8 clock delay      10100b - 5/2 clock delay      10101b - 21/8 clock delay      10110b - 16/4 clock delay      10111b - 23/8 clock delay      11000b - 3 clock delay      11001b - 25/8 clock delay      11010b - 13/4 clock delay      11011b - 27/8 clock delay</p>

Table continues on the next page...

## DDR register descriptions

Field	Function
	11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay
8-10 —	Reserved
11-15 WRLVL_START _2	Write leveling start time for DQS[2]. Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled. 00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START] 00001b - 1/8 clock delay 00010b - 1/4 clock delay 00011b - 3/8 clock delay 00100b - 1/2 clock delay 00101b - 5/8 clock delay 00110b - 3/4 clock delay 00111b - 7/8 clock delay 01000b - 1 clock delay 01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay 11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay
16-18 —	Reserved
19-23 WRLVL_START _3	Write leveling start time for DQS[3]. Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled. 00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START] 00001b - 1/8 clock delay 00010b - 1/4 clock delay 00011b - 3/8 clock delay 00100b - 1/2 clock delay 00101b - 5/8 clock delay 00110b - 3/4 clock delay 00111b - 7/8 clock delay 01000b - 1 clock delay

Table continues on the next page...

Field	Function
	01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay 11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay
24-26	Reserved
—	
27-31	Write leveling start time for DQS[4].
WRLVL_START_4	Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled. 00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START] 00001b - 1/8 clock delay 00010b - 1/4 clock delay 00011b - 3/8 clock delay 00100b - 1/2 clock delay 00101b - 5/8 clock delay 00110b - 3/4 clock delay 00111b - 7/8 c'lock delay 01000b - 1 clock delay 01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay

## DDR register descriptions

Field	Function
	11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay

## 18.4.28 DDR write leveling control 3 (DDR\_WRLVL\_CNTL\_3)

### 18.4.28.1 Offset

Register	Offset
DDR_WRLVL_CNTL_3	194h

### 18.4.28.2 Function

The DDR Write Leveling Control 3 register provides controls for write leveling. This register specifically defines the starting points for the individual data strobes.

Note: For each field WRLVL\_START\_n, a setting of 0000 is not recommended; it disables the per-byte write level start time selection. It is recommended to select proper individual delay for each byte lane based on board skews.

### 18.4.28.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			WRLVL_START_5				Reserved			WRLVL_START_6					
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			WRLVL_START_7				Reserved			WRLVL_START_8					
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.28.4 Fields

Field	Function
0-2 —	Reserved
3-7 WRLVL_START _5	<p>Write leveling start time for DQS[5].</p> <p>Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled.</p> <p>00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START]      00001b - 1/8 clock delay      00010b - 1/4 clock delay      00011b - 3/8 clock delay      00100b - 1/2 clock delay      00101b - 5/8 clock delay      00110b - 3/4 clock delay      00111b - 7/8 c'lock delay      01000b - 1 clock delay      01001b - 9/8 clock delay      01010b - 5/4 clock delay      01011b - 11/8 clock delay      01100b - 3/2 clock delay      01101b - 13/8 clock delay      01110b - 7/4 clock delay      01111b - 15/8 clock delay      10000b - 2 clock delay      10001b - 17/8 clock delay      10010b - 9/4 clock delay      10011b - 19/8 clock delay      10100b - 5/2 clock delay      10101b - 21/8 clock delay      10110b - 16/4 clock delay      10111b - 23/8 clock delay      11000b - 3 clock delay      11001b - 25/8 clock delay      11010b - 13/4 clock delay      11011b - 27/8 clock delay      11100b - 7/2 clock delay      11101b - 29/8 clock delay      11110b - 15/4 clock delay      11111b - 31/8 clock delay</p>
8-10 —	Reserved
11-15 WRLVL_START _6	<p>Write leveling start time for DQS[6].</p> <p>Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled.</p> <p>00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START]      00001b - 1/8 clock delay      00010b - 1/4 clock delay      00011b - 3/8 clock delay      00100b - 1/2 clock delay      00101b - 5/8 clock delay      00110b - 3/4 clock delay      00111b - 7/8 c'lock delay      01000b - 1 clock delay</p>

Table continues on the next page...

## DDR register descriptions

Field	Function
	01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay 11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay
16-18	Reserved
—	
19-23	Write leveling start time for DQS[7].
WRLVL_START — 7	Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled. 00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START] 00001b - 1/8 clock delay 00010b - 1/4 clock delay 00011b - 3/8 clock delay 00100b - 1/2 clock delay 00101b - 5/8 clock delay 00110b - 3/4 clock delay 00111b - 7/8 c'lock delay 01000b - 1 clock delay 01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay

Table continues on the next page...

Field	Function
	11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay
24-26	Reserved
—	
27-31	Write leveling start time for DQS[8].
WRLVL_START _8	Determines the value to use for the DQS_ADJUST for the first sample when write leveling is enabled. 00000b - Use value from DDR_WRLVL_CNTL[WRLVL_START] 00001b - 1/8 clock delay 00010b - 1/4 clock delay 00011b - 3/8 clock delay 00100b - 1/2 clock delay 00101b - 5/8 clock delay 00110b - 3/4 clock delay 00111b - 7/8 clock delay 01000b - 1 clock delay 01001b - 9/8 clock delay 01010b - 5/4 clock delay 01011b - 11/8 clock delay 01100b - 3/2 clock delay 01101b - 13/8 clock delay 01110b - 7/4 clock delay 01111b - 15/8 clock delay 10000b - 2 clock delay 10001b - 17/8 clock delay 10010b - 9/4 clock delay 10011b - 19/8 clock delay 10100b - 5/2 clock delay 10101b - 21/8 clock delay 10110b - 16/4 clock delay 10111b - 23/8 clock delay 11000b - 3 clock delay 11001b - 25/8 clock delay 11010b - 13/4 clock delay 11011b - 27/8 clock delay 11100b - 7/2 clock delay 11101b - 29/8 clock delay 11110b - 15/4 clock delay 11111b - 31/8 clock delay

## 18.4.29 DDR Register Control Words 3 (DDR\_SDRAM\_RCW\_3)

### 18.4.29.1 Offset

Register	Offset
DDR_SDRAM_RCW_3	1A0h

### 18.4.29.2 Function

The DDR Register Control Word 3 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 8-bit field represents the value that will be placed on MA[7:0], during register control word writes. This register is only used for DDR4 mode.

### 18.4.29.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW1X								RCW2X							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW3X								RCW4X							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.29.4 Fields

Field	Function
0-7 RCW1X	Register Control Word 1X. Represents the value that will be placed on MA[7:0] during writes to register control word 1X for DDR4.
8-15 RCW2X	Register Control Word 2X. Represents the value that will be placed on MA[7:0] during writes to register control word 2X for DDR4.
16-23 RCW3X	Register Control Word 3X. Represents the value that will be placed on MA[7:0] during writes to register control word 3X for DDR4.
24-31 RCW4X	Register Control Word 4X. Represents the value that will be placed on MA[7:0] during writes to register control word 4X for DDR4.

### 18.4.30 DDR Register Control Words 4 (DDR\_SDRAM\_RCW\_4)

### 18.4.30.1 Offset

Register	Offset
DDR_SDRAM_RCW_4	1A4h

### 18.4.30.2 Function

The DDR Register Control Word 4 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 8-bit field represents the value that will be placed on MA[7:0], during register control word writes. This register is only used for DDR4 mode.

### 18.4.30.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW5X								RCW6X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW7X								RCW8X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.30.4 Fields

Field	Function
0-7	Register Control Word 5X.
RCW5X	Represents the value that will be placed on MA[7:0] during writes to register control word 5X for DDR4.
8-15	Register Control Word 6X.
RCW6X	Represents the value that will be placed on MA[7:0] during writes to register control word 6X for DDR4.
16-23	Register Control Word 7X.
RCW7X	Represents the value that will be placed on MA[7:0] during writes to register control word 7X for DDR4.
24-31	Register Control Word 8X.
RCW8X	Represents the value that will be placed on MA[7:0] during writes to register control word 8X for DDR4.

## 18.4.31 DDR Register Control Words 5 (DDR\_SDRAM\_RCW\_5)

### 18.4.31.1 Offset

Register	Offset
DDR_SDRAM_RCW_5	1A8h

### 18.4.31.2 Function

The DDR Register Control Word 5 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 8-bit field represents the value that will be placed on MA[7:0], during register control word writes. This register is only used for DDR4 mode.

t

### 18.4.31.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW9X								RCW10X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW11X								RCW12X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.31.4 Fields

Field	Function
0-7 RCW9X	Register Control Word 9X. Represents the value that will be placed on MA[7:0] during writes to register control word 9X for DDR4.
8-15	Register Control Word 10X.

Table continues on the next page...

Field	Function
RCW10X	Represents the value that will be placed on MA[7:0] during writes to register control word 10X for DDR4.
16-23	Register Control Word 11X.
RCW11X	Represents the value that will be placed on MA[7:0] during writes to register control word 11X for DDR4.
24-31	Register Control Word 12X.
RCW12X	Represents the value that will be placed on MA[7:0] during writes to register control word 12X for DDR4.

## 18.4.32 DDR Register Control Words 6 (DDR\_SDRAM\_RCW\_6)

### 18.4.32.1 Offset

Register	Offset
DDR_SDRAM_RCW_6	1ACh

### 18.4.32.2 Function

The DDR Register Control Word 6 register should be programmed with the intended values of the register control words if DDR\_SDRAM\_CFG[RCW\_EN] is set. Each 8-bit field represents the value that will be placed on MA[7:0], during register control word writes. This register is only used for DDR4 mode.

### 18.4.32.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RCW13X								RCW14X							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW15X								SPARE_CNFG							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.32.4 Fields

Field	Function
0-7	Register Control Word 13X.
RCW13X	Represents the value that will be placed on MA[7:0] during writes to register control word 13X for DDR4.
8-15	Register Control Word 14X.
RCW14X	Represents the value that will be placed on MA[7:0] during writes to register control word 14X for DDR4.
16-23	Register Control Word 15X.
RCW15X	Represents the value that will be placed on MA[7:0] during writes to register control word 15X for DDR4.
24-31	Spare Config Bits.
SPARE_CNFG	This field is currently unused.

### 18.4.33 DDR SDRAM mode configuration 3 (DDR\_SDRAM\_MODE\_3)

#### 18.4.33.1 Offset

Register	Offset
DDR_SDRAM_MODE_3	200h

#### 18.4.33.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 1 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.33.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SDMODE															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.33.4 Fields

Field	Function
0-15 ESDMODE	<p>Extended SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM extended mode register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb of ESDMODE, which, in the big-endian convention shown here, corresponds to ESDMODE[15]. The msb of the SDRAM extended mode register value must be stored at ESDMODE[0]. The value programmed into this field is also used for writing MR1 during write leveling, although the bits specifically related to the write leveling scheme are handled automatically by the DDR controller. Even if DDR_SDRAM_CFG[B1] is set, this field is still used during write leveling. The write leveling enable bit should be cleared by software in this field.</p>
16-31 SDMODE	<p>SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM mode register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of SDMODE, which, in the big-endian convention shown here corresponds to SDMODE[15]. The msb of the SDRAM mode register value must be stored at SDMODE[0]. Because the memory controller forces SDMODE[7] to certain values depending on the state of the initialization sequence, (for resetting the SDRAM's DLL) the corresponding bits of this field are ignored by the memory controller. Note that SDMODE[7] is mapped to MA[8].</p>

### 18.4.34 DDR SDRAM mode configuration 4 (DDR\_SDRAM\_MODE\_4)

### 18.4.34.1 Offset

Register	Offset
DDR_SDRAM_MODE_4	204h

### 18.4.34.2 Function

The DDR SDRAM mode 4 configuration register sets the values loaded into the DDR's extended mode 2 and 3 registers. This register is used specifically for chip select 1 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.34.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ESDMODE3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.34.4 Fields

Field	Function
0-15 ESDMODE2	Extended SDRAM mode 2.  Specifies the initial value loaded into the DDR SDRAM extended 2 mode register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.  When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE2, which, in the big-endian convention shown here, corresponds to ESDMODE2[15]. The msb of the SDRAM extended mode 2 register value must be stored at ESDMODE2[0].
16-31 ESDMODE3	Extended SDRAM mode 3.  Specifies the initial value loaded into the DDR SDRAM extended 3 mode register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values of legal values is specified by the DDR SDRAM manufacturer.

Field	Function
	When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of ESDMODE3, which, in the big-endian convention shown here, corresponds to ESDMODE3[15]. The msb of the SDRAM extended mode 3 register value must be stored at ESDMODE3[0].

## 18.4.35 DDR SDRAM mode configuration 5 (DDR\_SDRAM\_MODE\_5)

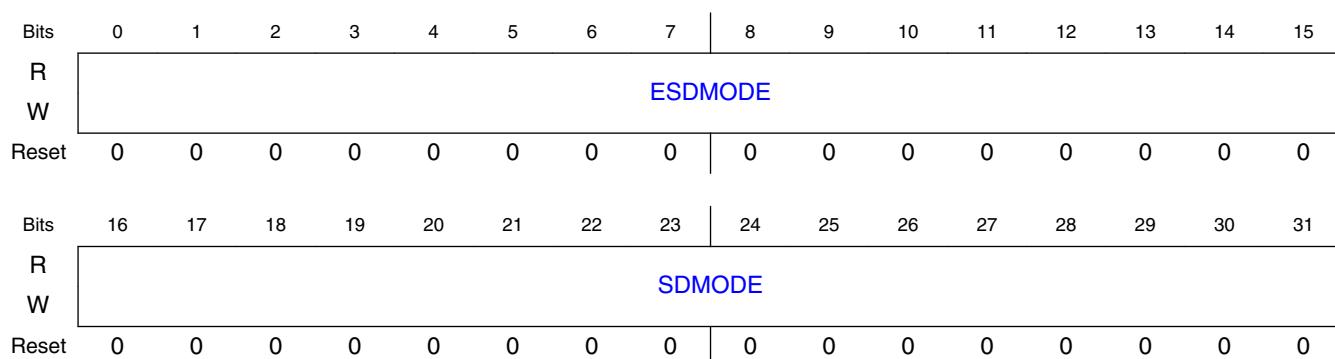
### 18.4.35.1 Offset

Register	Offset
DDR_SDRAM_MODE_5	208h

### 18.4.35.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 2 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.35.3 Diagram



### 18.4.35.4 Fields

Field	Function
0-15 ESDMODE	Extended SDRAM mode.  Specifies the initial value loaded into the DDR SDRAM extended mode register for chip select 2 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.  When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb of ESDMODE, which, in the big-endian convention, corresponds to ESDMODE[15]. The msb of the SDRAM extended mode register value must be stored at ESDMODE[0]. The value programmed into this field is also used for writing MR1 during write leveling, although the bits specifically related to the write leveling scheme are handled automatically by the DDR controller. Even if DDR_SDRAM_CFG[B1] is set, this field is still used during write leveling. The write leveling enable bit should be cleared by software in this field.
16-31 SDMODE	SDRAM mode.  Specifies the initial value loaded into the DDR SDRAM mode register for chip select 2 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values is specified by the DDR SDRAM manufacturer.  When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of SDMODE, which, in the big-endian convention, corresponds to SDMODE[15]. The msb of the SDRAM mode register value must be stored at SDMODE[0]. Because the memory controller forces SDMODE[7] to certain values depending on the state of the initialization sequence, (for resetting the SDRAM's DLL) the corresponding bits of this field are ignored by the memory controller. Note that SDMODE[7] is mapped to MA[8].

### 18.4.36 DDR SDRAM mode configuration 6 (DDR\_SDRAM\_MODE\_6)

#### 18.4.36.1 Offset

Register	Offset
DDR_SDRAM_MODE_6	20Ch

#### 18.4.36.2 Function

The DDR SDRAM mode 6 configuration register sets the values loaded into the DDR's extended mode 2 and 3 registers. This register is used specifically for chip select 2 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.36.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE2															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ESDMODE3															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.36.4 Fields

Field	Function
0-15 ESDMODE2	<p>Extended SDRAM mode 2.</p> <p>Specifies the initial value loaded into the DDR SDRAM extended 2 mode register for chip select 2 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE2, which, in the big-endian convention, corresponds to ESDMODE2[15]. The msb of the SDRAM extended mode 2 register value must be stored at ESDMODE2[0].</p>
16-31 ESDMODE3	<p>Extended SDRAM mode 3.</p> <p>Specifies the initial value loaded into the DDR SDRAM extended 3 mode register for chip select 2 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of ESDMODE3, which, in the big-endian convention, corresponds to ESDMODE3[15]. The msb of the SDRAM extended mode 3 register value must be stored at ESDMODE3[0].</p>

## 18.4.37 DDR SDRAM mode configuration 7 (DDR\_SDRAM\_MODE\_7)

### 18.4.37.1 Offset

Register	Offset
DDR_SDRAM_MODE_7	210h

### 18.4.37.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 3 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.37.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									SDMODE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.37.4 Fields

Field	Function
0-15 ESDMODE	<p>Extended SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM extended mode register for chip select 3 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb of ESDMODE, which, in the big-endian convention, corresponds to ESDMODE[15]. The msb of the SDRAM extended mode register value must be stored at ESDMODE[0]. The value programmed into this field is also used for writing MR1 during write leveling, although the bits specifically related to the write leveling scheme are handled automatically by the DDR controller. Even if DDR_SDRAM_CFG[B1] is set, this field is still used during write leveling. The write leveling enable bit should be cleared by software in this field.</p>
16-31 SDMODE	<p>SDRAM mode.</p> <p>Specifies the initial value loaded into the DDR SDRAM mode register for chip select 3 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values is specified by the DDR SDRAM manufacturer.</p> <p>When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of SDMODE, which, in the big-endian convention, corresponds to SDMODE[15]. The msb of the SDRAM mode register value must be stored at SDMODE[0]. Because the memory controller forces SDMODE[7] to certain values depending on the state of the initialization sequence, (for resetting the SDRAM's DLL) the corresponding bits of this field are ignored by the memory controller. Note that SDMODE[7] is mapped to MA[8].</p>

## 18.4.38 DDR SDRAM mode configuration 8 (DDR\_SDRAM\_MODE\_8)

### 18.4.38.1 Offset

Register	Offset
DDR_SDRAM_MODE_8	214h

### 18.4.38.2 Function

The DDR SDRAM mode 8 configuration register sets the values loaded into the DDR's extended mode 2 and 3 registers. This register is used specifically for chip select 3 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.38.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ESDMODE3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.38.4 Fields

Field	Function
0-15	Extended SDRAM mode 2.
ESDMODE2	Specifies the initial value loaded into the DDR SDRAM extended 2 mode register for chip select 3 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range and meaning of legal values is specified by the DDR SDRAM manufacturer.

*Table continues on the next page...*

## DDR register descriptions

Field	Function
	When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE2, which, in the big-endian convention, corresponds to ESDMODE2[15]. The msb of the SDRAM extended mode 2 register value must be stored at ESDMODE2[0].
16-31 ESDMODE3	Extended SDRAM mode 3.  Specifies the initial value loaded into the DDR SDRAM extended 3 mode register for chip select 3 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. The range of legal values of legal values is specified by the DDR SDRAM manufacturer.  When this value is driven onto the address bus (during DDR SDRAM initialization), MA[0] presents the lsb of ESDMODE3, which, in the big-endian convention, corresponds to ESDMODE3[15]. The msb of the SDRAM extended mode 3 register value must be stored at ESDMODE3[0].

## 18.4.39 DDR SDRAM mode configuration 9 (DDR\_SDRAM\_MODE\_9)

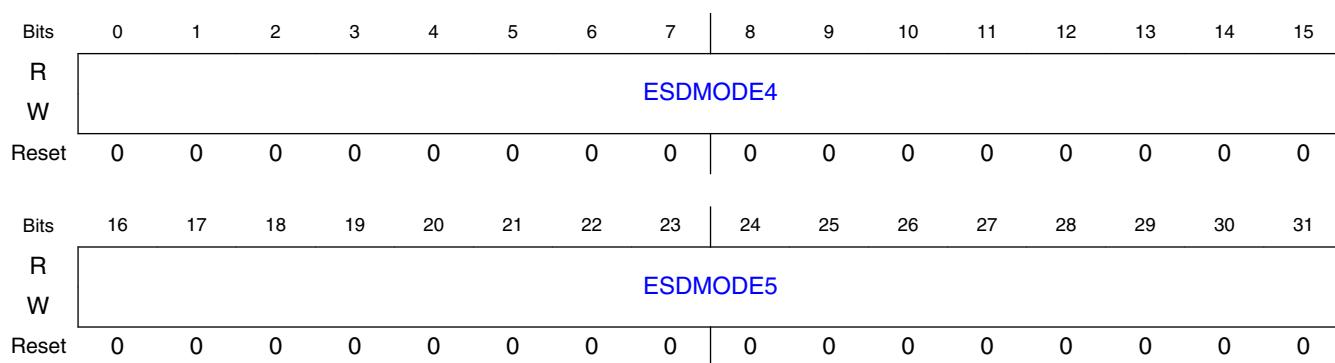
### 18.4.39.1 Offset

Register	Offset
DDR_SDRAM_MODE_9	220h

### 18.4.39.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers.

### 18.4.39.3 Diagram



### 18.4.39.4 Fields

Field	Function
0-15 ESDMODE4	Extended SDRAM mode 4. Specifies the initial value loaded into the DDR SDRAM MR4 register if using DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE4, which, in the big-endian convention, corresponds to ESDMODE4[15]. The msb of the SDRAM MR4 value must be stored at ESDMODE4[0].
16-31 ESDMODE5	Extended SDRAM mode 5. Specifies the initial value loaded into the DDR SDRAM MR5 register if using DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE5, which, in the big-endian convention, corresponds to ESDMODE5[15]. The msb of the SDRAM MR5 value must be stored at ESDMODE5[0].

### 18.4.40 DDR SDRAM mode configuration 10 (DDR\_SDRAM\_MODE\_10)

#### 18.4.40.1 Offset

Register	Offset
DDR_SDRAM_MODE_10	224h

#### 18.4.40.2 Function

The DDR SDRAM mode 10 configuration register sets the values loaded into the DDR's extended mode 6and 7 registers.

### 18.4.40.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE6							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ESDMODE7							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.40.4 Fields

Field	Function
0-15 ESDMODE6	Extended SDRAM mode 6. Specifies the initial value loaded into the DDR SDRAM MR6 register if using DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE6, which, in the big-endian convention, corresponds to ESDMODE6[15]. The msb of the SDRAM MR6 value must be stored at ESDMODE6[0].
16-31 ESDMODE7	Extended SDRAM mode 7. Specifies the initial value loaded into the DDR SDRAM MR7 register if using DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE7, which, in the big-endian convention, corresponds to ESDMODE7[15]. The msb of the SDRAM MR7 value must be stored at ESDMODE7[0].

## 18.4.41 DDR SDRAM mode configuration 11 (DDR\_SDRAM\_MODE\_11)

### 18.4.41.1 Offset

Register	Offset
DDR_SDRAM_MODE_1	228h

### 18.4.41.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 1 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.41.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE4							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ESDMODE5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.41.4 Fields

Field	Function
0-15 ESDMODE4	Extended SDRAM mode 4. Specifies the initial value loaded into the DDR SDRAM MR4 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE4, which, in the big-endian convention, corresponds to ESDMODE4[15]. The msb of the SDRAM MR4 value must be stored at ESDMODE4[0].
16-31 ESDMODE5	Extended SDRAM mode 5. Specifies the initial value loaded into the DDR SDRAM MR5 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE5, which, in the big-endian convention, corresponds to ESDMODE5[15]. The msb of the SDRAM MR5 value must be stored at ESDMODE5[0].

### 18.4.42 DDR SDRAM mode configuration 12 (DDR\_SDRAM\_MODE\_12)

### 18.4.42.1 Offset

Register	Offset
DDR_SDRAM_MODE_1 2	22Ch

### 18.4.42.2 Function

The DDR SDRAM mode 12 configuration register sets the values loaded into the DDR's extended mode 6 and 7 registers. This register is used specifically for chip select 1 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.42.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE6							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ESDMODE7							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.42.4 Fields

Field	Function
0-15 ESDMODE6	Extended SDRAM mode 6. Specifies the initial value loaded into the DDR SDRAM MR6 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE6, which, in the big-endian convention, corresponds to ESDMODE6[15]. The msb of the SDRAM MR6 value must be stored at ESDMODE6[0].
16-31 ESDMODE7	Extended SDRAM mode 7. Specifies the initial value loaded into the DDR SDRAM MR7 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto

Field	Function
	the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE7, which, in the big-endian convention, corresponds to ESDMODE7[15]. The msb of the SDRAM MR7 value must be stored at ESDMODE7[0].

## 18.4.43 DDR SDRAM mode configuration 13 (DDR\_SDRAM\_MODE\_13)

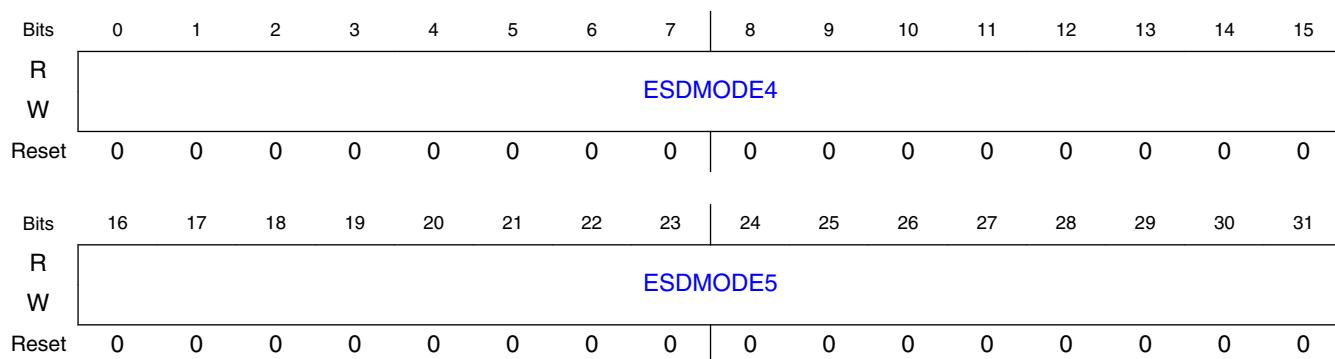
### 18.4.43.1 Offset

Register	Offset
DDR_SDRAM_MODE_13	230h

### 18.4.43.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 2 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.43.3 Diagram



### 18.4.43.4 Fields

Field	Function
0-15 ESDMODE4	Extended SDRAM mode 4. Specifies the initial value loaded into the DDR SDRAM MR4 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE4, which, in the big-endian convention corresponds to ESDMODE4[15]. The msb of the SDRAM MR4 value must be stored at ESDMODE4[0].
16-31 ESDMODE5	Extended SDRAM mode 5. Specifies the initial value loaded into the DDR SDRAM MR5 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE5, which, in the big-endian convention corresponds to ESDMODE5[15]. The msb of the SDRAM MR5 value must be stored at ESDMODE5[0].

### 18.4.44 DDR SDRAM mode configuration 14 (DDR\_SDRAM\_MODE\_14)

#### 18.4.44.1 Offset

Register	Offset
DDR_SDRAM_MODE_14	234h

#### 18.4.44.2 Function

The DDR SDRAM mode 14 configuration register sets the values loaded into the DDR's extended mode 6 and 7 registers. This register is used specifically for chip select 2 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.44.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE6															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ESDMODE7															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.44.4 Fields

Field	Function
0-15 ESDMODE6	Extended SDRAM mode 6. Specifies the initial value loaded into the DDR SDRAM MR6 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE6, which, in the big-endian convention, corresponds to ESDMODE6[15]. The msb of the SDRAM MR6 value must be stored at ESDMODE6[0].
16-31 ESDMODE7	Extended SDRAM mode 7. Specifies the initial value loaded into the DDR SDRAM MR7 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE7, which, in the big-endian convention, corresponds to ESDMODE7[15]. The msb of the SDRAM MR7 value must be stored at ESDMODE7[0].

## 18.4.45 DDR SDRAM mode configuration 15 (DDR\_SDRAM\_MODE\_15)

### 18.4.45.1 Offset

Register	Offset
DDR_SDRAM_MODE_15	238h

### 18.4.45.2 Function

The DDR SDRAM mode configuration register sets the values loaded into the DDR's mode registers. This register is used specifically for chip select 3 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.45.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ESDMODE4							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ESDMODE5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.45.4 Fields

Field	Function
0-15 ESDMODE4	Extended SDRAM mode 4. Specifies the initial value loaded into the DDR SDRAM MR4 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE4, which, in the big-endian convention, corresponds to ESDMODE4[15]. The msb of the SDRAM MR4 value must be stored at ESDMODE4[0].
16-31 ESDMODE5	Extended SDRAM mode 5. Specifies the initial value loaded into the DDR SDRAM MR5 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE5, which, in the big-endian convention, corresponds to ESDMODE5[15]. The msb of the SDRAM MR5 value must be stored at ESDMODE5[0].

## 18.4.46 DDR SDRAM mode configuration 16 (DDR\_SDRAM\_MODE\_16)

### 18.4.46.1 Offset

Register	Offset
DDR_SDRAM_MODE_16	23Ch

### 18.4.46.2 Function

The DDR SDRAM mode 16 configuration register sets the values loaded into the DDR's extended mode 6 and 7 registers. This register is used specifically for chip select 3 if DDR\_SDRAM\_CFG\_2[UNQ\_MRS\_EN] is set.

### 18.4.46.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ESDMODE6															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ESDMODE7															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.46.4 Fields

Field	Function
0-15 ESDMODE6	Extended SDRAM mode 6. Specifies the initial value loaded into the DDR SDRAM MR6 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE6, which, in the big-endian convention, corresponds to ESDMODE6[15]. The msb of the SDRAM MR6 value must be stored at ESDMODE6[0].

Table continues on the next page...

## DDR register descriptions

Field	Function
16-31 ESDMODE7	Extended SDRAM mode 7. Specifies the initial value loaded into the DDR SDRAM MR7 register for chip select 1 if DDR_SDRAM_CFG_2[UNQ_MRS_EN] is set. This is only used with DDR4 memories. The range and meaning of legal values is specified by the DDR SDRAM manufacturer. When this value is driven onto the address bus (during the DDR SDRAM initialization sequence), MA[0] presents the lsb bit of ESDMODE7, which, in the big-endian convention, corresponds to ESDMODE7[15]. The msb of the SDRAM MR7 value must be stored at ESDMODE7[0].

## 18.4.47 DDR SDRAM timing configuration 8 (TIMING\_CFG\_8)

### 18.4.47.1 Offset

Register	Offset
TIMING_CFG_8	250h

### 18.4.47.2 Function

The DDR SDRAM timing configuration 8 register provides additional timing fields required to support DDR4 memories.

### 18.4.47.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RWT_BG				WRT_BG				RRT_BG				WWT_BG			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ACTTOACT_BG				WRTORD_BG				Reserved				PRE_ALL_REC			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.47.4 Fields

Field	Function
0-3 RWT_BG	<p>Read-to-write turnaround for same chip select and same bank group.</p> <p>Specifies how many cycles will be added between a read to write turnaround for transactions to the same bank group. If a value of 0000 is chosen, then the DDR controller will use the value used for transactions to different chip selects, as defined in TIMING_CFG_0[RWT]. This field can be used to improve performance when operating in burst-chop mode by forcing transactions to the same chip select to use extra cycles, while transaction to different chip selects can utilize the tri-state time on the DRAM interface. Regardless of the value that is set in this field, the value defined by TIMING_CFG_0[RWT] will also be met before issuing a write command. This field should be set to 0000 for DDR3.</p> <ul style="list-style-type: none"> <li>0000b - Default</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
4-7 WRT_BG	<p>Write-to-read turnaround for same chip select and same bank group.</p> <p>Specifies how many cycles will be added between a write to read turnaround for transactions to the same bank group. If a value of 0000 is chosen, then the DDR controller will use the value used for transactions to different chip selects, as defined in TIMING_CFG_0[WRT]. This field can be used to improve performance when operating in burst-chop mode by forcing transactions to the same chip select to use extra cycles, while transaction to different chip selects can utilize the tri-state time on the DRAM interface. Regardless of the value that is set in this field, the value defined by TIMING_CFG_0[WRT] will also be met before issuing a read command. This field should be programmed to 0000 for DDR3 mode.</p> <ul style="list-style-type: none"> <li>0000b - Default</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> <li>0100b - 4 clocks</li> <li>0101b - 5 clocks</li> <li>0110b - 6 clocks</li> <li>0111b - 7 clocks</li> <li>1000b - 8 clocks</li> <li>1001b - 9 clocks</li> <li>1010b - 10 clocks</li> <li>1011b - 11 clocks</li> <li>1100b - 12 clocks</li> <li>1101b - 13 clocks</li> <li>1110b - 14 clocks</li> <li>1111b - 15 clocks</li> </ul>
8-11 RRT_BG	Read-to-read turnaround for same chip select and same bank group.

*Table continues on the next page...*

## DDR register descriptions

Field	Function
	<p>Specifies how many cycles will be added between reads to the same bank group. If a value of 0000 is chosen, then 2 cycles will be required between read commands to the same chip select if 4-beat bursts are used (4 cycles will be required if 8-beat bursts are used). Note that fixed 4-beat bursts are not supported. However, this field may be used to add extra cycles when burst-chop mode is used, and the DDR controller must wait 4 cycles for read-to-read transactions to the same chip select. This field should be programmed to 0000 for DDR3 mode.</p> <ul style="list-style-type: none"> <li>0000b - BL/2 clocks</li> <li>0001b - BL/2 + 1 clock</li> <li>0010b - BL/2 + 2 clocks</li> <li>0011b - BL/2 + 3 clocks</li> <li>0100b - BL/2 + 4 clocks</li> <li>0101b - BL/2 + 5 clocks</li> <li>0110b - BL/2 + 6 clocks</li> <li>0111b - BL/2 + 7 clocks</li> <li>1000b - BL/2 + 8 clocks</li> <li>1001b - BL/2 + 9 clocks</li> <li>1010b - BL/2 + 10 clocks</li> <li>1011b - BL/2 + 11 clocks</li> <li>1100b - BL/2 + 12 clocks</li> <li>1101b - BL/2 + 13 clocks</li> <li>1110b - BL/2 + 14 clocks</li> <li>1111b - BL/2 + 15 clocks</li> </ul>
12-15 WWT_BG	<p>Write-to-write turnaround for same chip select and same bank group.</p> <p>Specifies how many cycles will be added between writes to the same bank group. If a value of 0000 is chosen, then 2 cycles will be required between write commands to the same bank group if 4-beat bursts are used (4 cycles will be required if 8-beat bursts are used). Note that fixed 4-beat bursts are not supported. However, this field may be used to add extra cycles when burst-chop mode is used, and the DDR controller must wait 4 cycles for write-to-write transactions to the same chip select. This field should be programmed to 0000 for DDR3 mode.</p> <ul style="list-style-type: none"> <li>0000b - BL/2 clocks</li> <li>0001b - BL/2 + 1 clock</li> <li>0010b - BL/2 + 2 clocks</li> <li>0011b - BL/2 + 3 clocks</li> <li>0100b - BL/2 + 4 clocks</li> <li>0101b - BL/2 + 5 clocks</li> <li>0110b - BL/2 + 6 clocks</li> <li>0111b - BL/2 + 7 clocks</li> <li>1000b - BL/2 + 8 clocks</li> <li>1001b - BL/2 + 9 clocks</li> <li>1010b - BL/2 + 10 clocks</li> <li>1011b - BL/2 + 11 clocks</li> <li>1100b - BL/2 + 12 clocks</li> <li>1101b - BL/2 + 13 clocks</li> <li>1110b - BL/2 + 14 clocks</li> <li>1111b - BL/2 + 15 clocks</li> </ul>
16-19 ACTTOACT_BG	<p>Activate-To-Activate Same Bank Group.</p> <p>Activate-to-activate interval for the same bank group(<math>t_{RRD\_L}</math>). Number of clock cycles from an activate command until another activate command is allowed for a different logical bank in the same physical bank (chip select). Bank groups are only used for DDR4. This field should be programmed to 0000 for DDR3 mode.</p> <ul style="list-style-type: none"> <li>0000b - ACTTOACT_BG is unused</li> <li>0001b - 1 clock</li> <li>0010b - 2 clocks</li> <li>0011b - 3 clocks</li> </ul>

Table continues on the next page...

Field	Function
	0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
20-23 WRTORD_BG	Write-To-Read Same Bank Group.  Last write data pair to read command issue interval for the same bank group( $t_{WTR\_L}$ ). Number of clock cycles between the last write data pair and the subsequent read command to the same physical bank. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this field needs to be programmed to ( $t_{WTR} + 2$ cycles). Bank groups are only used for DDR4. This field should be programmed to 0000 for DDR3 mode.  0000b - WRTORD_BG timing is unused 0001b - 1 clock 0010b - 2 clocks 0011b - 3 clocks 0100b - 4 clocks 0101b - 5 clocks 0110b - 6 clocks 0111b - 7 clocks 1000b - 8 clocks 1001b - 9 clocks 1010b - 10 clocks 1011b - 11 clocks 1100b - 12 clocks 1101b - 13 clocks 1110b - 14 clocks 1111b - 15 clocks
24-26 —	Reserved
27-31 PRE_ALL_REC	Precharge all-to-activate interval.  Determines the number of clock cycles from a precharge all command until an activate or refresh command is allowed.  00000b - Use TIMING_CFG_1[PRETOACT] 00001b - 1 clock 00010b - 2 clocks 00011b - 3 clocks 00100b - 4 clocks 00101b - 5 clocks 00110b - 6 clocks 00111b - 7 clocks 01000b - 8 clocks 01001b - 9 clocks 01010b - 10 clocks 01011b - 11 clocks 01100b - 12 clocks 01101b - 13 clocks 01110b - 14 clocks 01111b - 15 clocks

## DDR register descriptions

Field	Function
	10000b - 16 clocks
	10001b - 17 clocks
	10010b - 18 clocks
	10011b - 19 clocks
	10100b - 20 clocks
	10101b - 21 clocks
	10110b - 22 clocks
	10111b - 23 clocks
	11000b - 24 clocks
	11001b - 25 clocks
	11010b - 26 clocks
	11011b - 27 clocks
	11100b - 28 clocks
	11101b - 29 clocks
	11110b - 30 clocks
	11111b - 31 clocks

## 18.4.48 DDR SDRAM control configuration 3 (DDR\_SDRAM\_CF\_G\_3)

### 18.4.48.1 Offset

Register	Offset
DDR_SDRAM_CFG_3	260h

### 18.4.48.2 Function

The DDR SDRAM control configuration register 3 provides more control configuration for the DDR controller.

### 18.4.48.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DDRC_RST	ECC_FIX_EN	Reserved		ECC_SCRUB_INT				Reserved				Reserved	Reserved	Reserved	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		DM_CFG	Reserved		REF_MOD_E			Reserved	Reserved	Reserved		Reserved	Reserved	Reserved	DIS_MRS_PAR
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.48.4 Fields

Field	Function
0 DDRC_RST	<p>DDR controller reset.</p> <p>This bit should be set by software if the DDR controller should be reset. This will reset all state machines and internal FIFOs. This bit will not reset the DDR controller's configuration registers. Before setting this bit, DDR_SDRAM_CFG[MEM_EN] should be cleared. Note that this bit is self-clearing. Software should poll for this bit to clear before re-enabling the DDR controller.</p> <p>0b - DDR controller is operating in normal mode. 1b - DDR controller is reset.</p>
1 ECC_FIX_EN	<p>ECC fixing enable.</p> <p>The DDR controller supports ECC fixing in memory. In this mode, the DDR controller will automatically fix any detected single-bit errors by issuing a new transaction to read the address with the failing bit, correcting the bit, and writing the data back to memory. The single-bit error will still be counted in the ERR_SBE register for this case, but the controller will automatically fix the error. Note that during the 'read back', the single-bit error will not be double counted in the ERR_SBE register. In addition, the DDR controller will periodically issue a read to memory at the interval defined by ECC_SCRUB_INT. If a single-bit error is detected during a periodic read, it will be fixed. In this case, the error will be reported as an SSBE in the ERR_SBE register. If a multi-bit error is detected, then it will be reported in the ERR_DETECT register. Also note that if a subsequent single-bit error is detected at the same address while a first error is being fixed, then the second error will not be reported. Also, after a first SBE is detected, no other SBEs will be fixed until the first SBE has been fixed in memory. This bit should only be set if DDR_SDRAM_CFG[ECC_EN] is also set.</p> <p><b>NOTE:</b> Scrubbing cannot be enabled until the controller has cleared DDR_SDRAM_CFG_2[D_INIT].</p> <p>0b - ECC scrubbing is disabled. 1b - ECC scrubbing is enabled.</p>

Table continues on the next page...

## DDR register descriptions

Field	Function
2-3 —	Reserved
4-7 ECC_SCRUB_INTERVAL	<p>ECC scrubbing interval.</p> <p>This field defines how frequent the DDR controller will inject reads to test ECC if ECC_FIX_EN is set. Note that reads will only be injected immediately after a refresh sequence. If a single-bit error is detected, then the controller will fix the error in memory. If a multi-bit error is detected, then the controller will report the error in the ERR_DETECT register. When issuing reads, the DDR controller will move sequentially throughout the DRAM address space by automatically incrementing an internal address counter.</p> <p>Note that the maximum amount memory supported would be 4 ranks of 16 Gbit x8 devices utilizing a 64-bit bus, which provides a total of 64 GBytes of memory. Since 64-bytes of data are scrubbed for each read, this will require 1G total reads to scrub the entire memory contents. Assuming a refresh interval of 7.8 microseconds, this will take approximately 2.35 hours to cycle through the entire memory array. If less memory is used, then this time will take less. The scrubbing interval can be increased to improve performance so the reads are not issued as frequently.</p> <p><b>NOTE:</b> Scrubbing cannot be enabled until after the controller has cleared DDR_SDRAM_CFG_2[D_INIT].</p> <ul style="list-style-type: none"> <li>0000b - Periodic reads for ECC scrubbing will not be issued.</li> <li>0001b - A read will be issued every refresh sequence.</li> <li>0010b - A read will be issued every 2 refresh sequences.</li> <li>0011b - A read will be issued every 4 refresh sequences.</li> <li>0100b - A read will be issued every 8 refresh sequences.</li> <li>0101b - A read will be issued every 16 refresh sequences.</li> <li>0110b - A read will be issued every 32 refresh sequences.</li> <li>0111b - A read will be issued every 64 refresh sequences.</li> <li>1000b - A read will be issued every 128 refresh sequences.</li> <li>1001b - A read will be issued every 256 refresh sequences.</li> <li>1010b - A read will be issued every 512 refresh sequences.</li> <li>1011b - A read will be issued every 1024 refresh sequences.</li> <li>1100b - A read will be issued every 2048 refresh sequences.</li> <li>1101b - A read will be issued every 4096 refresh sequences.</li> <li>1110b - A read will be issued every 8192 refresh sequences.</li> <li>1111b - A read will be issued every 16,384 refresh sequences.</li> </ul>
8-11 —	Reserved
12 —	Reserved
13 —	Reserved
14-15 —	Reserved
16-17 —	Reserved
18-19 DM_CFG	<p>Data mask config.</p> <p>Determines how the MDM pins will be utilized for the DDR controller.</p> <ul style="list-style-type: none"> <li>00b - Normal data masks will be used based on DDR_SDRAM_CFG[SDRAM_TYPE].</li> <li>01b - Reserved.</li> <li>10b - Data bus inversion (DBI) will be used by utilizing the MDM pins.</li> <li>11b - Neither data mask or data bus inversion will be used. The MDM pins will be held low for DDR3 by the controller. The MDM pins will be held high for DDR4 by the controller.</li> </ul>

Table continues on the next page...

Field	Function
20-21 —	Reserved
22-23 REF_MODE	<p>Refresh Mode.</p> <p>This field allows programming for fine granularity refresh defined by DDR4 specifications. The fine granularity refresh mode is supported by allowing full programmability of the DDR_SDRAM_CFG_2[NUM_PR] and DDR_SDRAM_INTERVAL[REFINT] timings. However, this field is required to notify the controller how many refreshes to issue when exiting self refresh. If fine granularity refresh mode will be used, then it must be enabled after the DDR controller has been enabled, per the DRAM requirement to use 1x mode during initial MRS commands. Therefore, software must issue the required MRS commands via DDR_SDRAM_MD_CNTL register to allow the DRAM to enter fine granularity refresh mode. On-the-fly fine granularity refresh mode is not supported.</p> <p>00b - Fine granularity refresh disabled. 01b - 2x fine granularity refresh mode. 10b - 4x fine granularity refresh mode. 11b - Reserved</p>
24 —	Reserved
25 —	Reserved
26-27 —	Reserved
28 —	Reserved
29 —	Reserved
30 —	Reserved
31 DIS_MRS_PAR	<p>Disable MRS on Parity Error.</p> <p>This bit controls the automatic MR command that may be issued upon a parity error. This bit should be set if using parity.</p> <p>If using parity with discrete devices, MR5 should be set to automatically re-enable parity checking after an error.</p> <p>0b - Issue an MR command to clear the parity error in the DRAM when a parity error is observed. If parity is enabled when using DDR4 memories, then this bit must be set. In this case, software is responsible for clearing the DRAM's mode register parity error bit if required. However, the DRAM should be configured for persistent parity errors by setting MR5[A9] when initializing the DDR controller to program the DRAM Mode Registers. This will ensure that the DDR4 DRAM will continue to check parity errors until software can clear a previous parity error.</p> <p>1b - Do not issue an MR command to clear the parity error on the DRAM. Software is responsible for clearing parity errors if required via the DDR_SDRAM_MD_CNTL register.</p>

#### 18.4.49 DQ mapping register 0 (DDR\_DQ\_MAP0)

### 18.4.49.1 Offset

Register	Offset
DDR_DQ_MAP0	400h

### 18.4.49.2 Function

DQ\_MAP0-DQ\_MAP3 should be programmed to reflect how the DQ bits are swizzled on the DIMMs. Note that if a board has further swizzling within the nibbles of the DQ signals, then that needs to be accounted for when programming these fields. In addition, if multiple modules are used in the system, they must use the same raw card design (such that the swizzling is identical between DIMMs).

For each of the DQ mapping fields, the encodings are shown in the accompanying table. These bitfields apply to nibbles of data; therefore, they must normally be set in pairs. (For example, DQ\_0\_3 and DQ\_4\_7 must be set together.)

**Table 18-7. Bitfield settings for each byte lane**

Setting	DQ0	DQ1	DQ2	DQ3
	DQ4	DQ5	DQ6	DQ7
0x00	0	1	2	3
0x01	0	1	2	3
0x02	0	1	3	2
0x03	0	2	1	3
0x04	0	2	3	1
0x05	0	3	1	2
0x06	0	3	2	1
0x07	1	0	2	3
0x08	1	0	3	2
0x09	1	2	0	3
0x0a	1	2	3	0
0x0b	1	3	0	2
0x0c	1	3	2	0
0x0d	2	0	1	3
0x0e	2	0	3	1
0x0f	2	1	0	3
0x10	2	1	3	0
0x11	2	3	0	1
0x12	2	3	1	0

*Table continues on the next page...*

**Table 18-7. Bitfield settings for each byte lane (continued)**

Setting	DQ0	DQ1	DQ2	DQ3
	DQ4	DQ5	DQ6	DQ7
0x13	3	0	1	2
0x14	3	0	2	1
0x15	3	1	0	2
0x16	3	1	2	0
0x17	3	2	0	1
0x18	3	2	1	0
0x21	4	5	6	7
0x22	4	5	7	6
0x23	4	6	5	7
0x24	4	6	7	5
0x25	4	7	5	6
0x26	4	7	6	5
0x27	5	4	6	7
0x28	5	4	7	6
0x29	5	6	4	7
0x2a	5	6	7	4
0x2b	5	7	4	6
0x2c	5	7	6	4
0x2d	6	4	5	7
0x2e	6	4	7	5
0x2f	6	5	4	7
0x30	6	5	7	4
0x31	6	7	4	5
0x32	6	7	5	4
0x33	7	4	5	6
0x34	7	4	6	5
0x35	7	5	4	6
0x36	7	5	6	4
0x37	7	6	4	5
0x38	7	6	5	4

### 18.4.49.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.49.4 Fields

Field	Function
0-5 DQ_0_3	DQ[0:3] Mapping. See <a href="#">Table 18-7</a> and the text describing this register for settings.
6-11 DQ_4_7	DQ[4:7] Mapping. See <a href="#">Table 18-7</a> and the text describing this register for settings.
12-17 DQ_8_11	DQ[8:11] Mapping. See <a href="#">Table 18-7</a> and the text describing this register for settings.
18-23 DQ_12_15	DQ[12:15] Mapping. See <a href="#">Table 18-7</a> and the text describing this register for settings.
24-29 DQ_16_19	DQ[16:19] Mapping. See <a href="#">Table 18-7</a> and the text describing this register for settings.
30-31 RESERVED	Reserved These bits are writeable, but they are unused.

## 18.4.50 DQ mapping register 1 (DDR\_DQ\_MAP1)

### 18.4.50.1 Offset

Register	Offset
DDR_DQ_MAP1	404h

### 18.4.50.2 Function

DQ\_MAP0-DQ\_MAP3 should be programmed to reflect how the DQ bits are swizzled on the DIMMs. See further details at [DQ mapping register 0 \(DDR\\_DQ\\_MAP0\)](#).

### 18.4.50.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DQ_20_23							DQ_24_27							DQ_28_31	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	DQ_28_31				DQ_32_35				DQ_36_39				RESERVE			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.50.4 Fields

Field	Function
0-5	DQ[20:23] Mapping.
DQ_20_23	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
6-11	DQ[24:27] Mapping.
DQ_24_27	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
12-17	DQ[28:31] Mapping.
DQ_28_31	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
18-23	DQ[32:35] Mapping.
DQ_32_35	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
24-29	DQ[36:39] Mapping.
DQ_36_39	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
30-31	Reserved
RESERVED	These bits are writeable, but they are unused.

## 18.4.51 DQ mapping register 2 (DDR\_DQ\_MAP2)

### 18.4.51.1 Offset

Register	Offset
DDR_DQ_MAP2	408h

### 18.4.51.2 Function

DQ\_MAP0-DQ\_MAP3 should be programmed to reflect how the DQ bits are swizzled on the DIMMs. See further details at [DQ mapping register 0 \(DDR\\_DQ\\_MAP0\)](#).

### 18.4.51.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DQ_40_43						DQ_44_47						DQ_48_51			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	DQ_48_51				DQ_52_55				DQ_56_59				RESERVE D			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.51.4 Fields

Field	Function
0-5	DQ[40:43] Mapping.
DQ_40_43	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
6-11	DQ[44:47] Mapping.
DQ_44_47	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
12-17	DQ[48:51] Mapping.
DQ_48_51	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
18-23	DQ[52:55] Mapping.

Table continues on the next page...

Field	Function
DQ_52_55	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
24-29	DQ[56:59] Mapping.
DQ_56_59	See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
30-31	Reserved
RESERVED	These bits are writeable, but they are unused.

## 18.4.52 DQ mapping register 3 (DDR\_DQ\_MAP3)

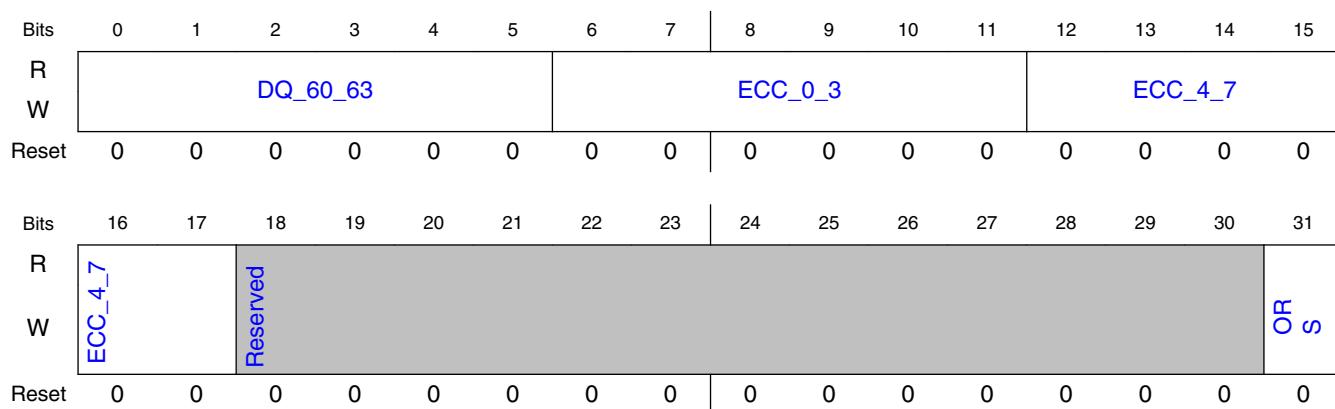
### 18.4.52.1 Offset

Register	Offset
DDR_DQ_MAP3	40Ch

### 18.4.52.2 Function

DQ\_MAP0-DQ\_MAP3 should be programmed to reflect how the DQ bits are swizzled on the DIMMs. See further details at [DQ mapping register 0 \(DDR\\_DQ\\_MAP0\)](#).

### 18.4.52.3 Diagram



## 18.4.52.4 Fields

Field	Function
0-5 DQ_60_63	DQ[60:63] Mapping. See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
6-11 ECC_0_3	ECC[0:3] Mapping. See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
12-17 ECC_4_7	ECC[4:7] Mapping. See the table and description in <a href="#">DQ mapping register 0 (DDR_DQ_MAP0)</a> for settings.
18-30 —	Reserved
31 ORS	Odd rank swizzle. If this bit is cleared, then CS1 and CS3 follow the same swizzling as CS0 and CS2. If this bit is set, then CS1 and CS3 are further swizzled by swapping bits 0 vs 1, 2 vs 3, 4 vs 5, and 6 vs 7 within each byte.

## 18.4.53 DDR Debug Status Register 1 (DDRDSR\_1)

### 18.4.53.1 Offset

Register	Offset
DDRDSR_1	B20h

### 18.4.53.2 Function

The DDRDSR\_1 register contains the current settings of the driver impedance for the command/control and data.

### 18.4.53.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CZ								DZ							
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

### 18.4.53.4 Fields

Field	Function
0-15	Reserved
—	
16	Command Impedance.
CZ	Current setting of driver command impedance
17-23	Reserved
—	
24	Data Impedance.
DZ	Current setting of driver data impedance
25-31	Reserved
—	

### 18.4.54 DDR Debug Status Register 2 (DDRDSR\_2)

#### 18.4.54.1 Offset

Register	Offset
DDRDSR_2	B24h

## 18.4.54.2 Function

The DDRDSR\_2 register contains the current settings of the driver impedance for the DDR drivers for clocks.

## 18.4.54.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CLKZ	Reserved														
W	CLKZ															
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved														RPD_ST	RPD_END
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.54.4 Fields

Field	Function
0 CLKZ	Clock Impedance. Current setting of driver clock impedance
1-29 —	Reserved
30 RPD_ST	Rapid clear of memory start. See <a href="#">DDR Rapid Clear of Memory</a> for more information. 0b - The rapid clear of memory function has not been started. 1b - The rapid clear of memory function has started. During the rapid clear of memory, writes to the DDR memory mapped registers will not affect the register contents. This bit is cleared by software writing a 1.
31 RPD_END	Rapid clear of memory end. See <a href="#">DDR Rapid Clear of Memory</a> for more information. 0b - The rapid clear of memory function has not ended. 1b - The rapid clear of memory function has completed. This bit is cleared by software writing a 1.

## 18.4.55 DDR Control Driver Register 1 (DDRCDR\_1)

### 18.4.55.1 Offset

Register	Offset
DDRCDR_1	B28h

### 18.4.55.2 Function

DDRCDR\_1 sets the hardware DDR driver calibration enable, the ODT termination value for IOs, and the software override enables for address/command and data bus signals. The combined DDRCDR\_1[ODT] and DDRCDR\_2[ODT] set the on-die-termination values in the memory controller for the data bus signals. Hardware DDR driver calibration must be enabled by setting DDRCDR\_1[DHC\_EN]. MDIC pins are required to be connected to proper calibration resistors. The proper value and connection of the MDIC resistor are specified in the corresponding device data sheet document.

The global half-strength override field DDR\_SDRAM\_CFG[HSE] or software overrides in DDRCDR\_1 or DDRCDR\_2 registers determine whether the DDR drivers calibrate to full-strength or half-strength. The software overrides in DDRCDR\_1 or DDRCDR\_2 registers take precedence over the DDR\_SDRAM\_CFG[HSE] setting.

#### NOTE

All driver calibration related registers should be set 500  $\mu$ s before the DDR controller is enabled (that is, before DDR\_SDRAM\_CFG[MEM\_EN] is set).

This table lists the valid impedance override values. Note that the drivers may be calibrated to either full-strength or half-strength.

**Table 18-8. Valid Impedance Override Values**

Driver impedance	Impedance Override Value
Highest	0
Lowest	1

### 18.4.55.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DHC_EN	Reserved								ODT			DSO_C_EN	DSO_D_EN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	DSO_CZ	Reserved								DSO_DZ	Reserved					
W									0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.55.4 Fields

Field	Function
0 DHC_EN	Driver Hardware Calibration Enable. DDR driver hardware compensation enable. If this bit is set to enable automatic hardware compensation, then 200 microseconds should pass after setting this bit, and before setting DDR_SDRAM_CFG[MEM_EN].
1-11 —	Reserved These bits are writeable, but they are unused.
12-13 ODT	On-Die-Termination. ODT termination value for IOs. This is combined with DDRCDR_2[ODT] to determine the termination value. Below is the termination based on concatenating these two fields. Note that the order of concatenation is from left to right DDRCDR_1[ODT], DDRCDR_2[ODT] Bits [0:0] of the following values are defined at <a href="#">DDRCDR_2.ODT</a> . 000b - Termination is disabled 001b - 120 Ω (DDR3) 100 Ω (DDR4) 010b - 200 Ω (DDR3) 120 Ω (DDR4) 011b - 75 Ω (DDR3) 80 Ω (DDR4) 100b - Reserved (DDR3) 60 Ω (DDR4) 101b - 60 Ω (DDR3) 40 Ω (DDR4) 110b - Reserved (DDR3) 50 Ω (DDR4) 111b - 46 Ω (DDR3) 30 Ω (DDR4)
14 DSO_C_EN	Override Enable for Command Impedance. Driver software override enable for address/command
15 DSO_D_EN	Override Enable for Data Impedance. Driver software override enable for data

Table continues on the next page...

Field	Function
16 DSO_CZ	Override for Command Impedance. DDR driver software command impedance override
17-23 —	Reserved These bits are writeable, but they are unused.
24 DSO_DZ	Override For Data Impedance. Driver software data impedance override
25-31 —	Reserved These bits are writeable, but they are unused.

## 18.4.56 DDR Control Driver Register 2 (DDRCDR\_2)

### 18.4.56.1 Offset

Register	Offset
DDRCDR_2	B2Ch

### 18.4.56.2 Function

The DDRCDR\_2 sets the driver software override enable for clocks, and the DDR clocks driver P/N impedance.

### 18.4.56.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DSO_CLK_EN	Reserved			DSO_CLKZ	Reserved								Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	VREF_OVRD_EN	Reserved	VREF_OVRD_VAL						VREF_TRAIN_EN	VREF_DRAM_RANGE	Reserved					ODT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.56.4 Fields

Field	Function
0	Override Enable for Clock Impedance.
DSO_CLK_EN	Driver software override enable for clocks
1-3 —	Reserved
4	Override Value for Clock Impedance.
DSO_CLKZ	Driver software clocks impedance override
5-11 —	Reserved
12-15 —	Reserved
16	Internal VRef Override Enable.
VREF_OVRD_EN	If DDR4 mode is used, then an internal VRef is generated and used for the data bus. By default, this internal VRef value will be trained. However, the training can be disabled by this override.  0b - Internal VRef will be trained if DDR4 mode is used. 1b - Internal VRef will be defined by the VREF_OVRD_VAL field of this register.
17 —	Reserved
18-23	Internal VRef Override Value.

Table continues on the next page...

Field	Function
VREF_OVRD_V AL	This field defines the override value to use for the internal VRef if VREF_OVRD_EN is set. The values below are targeted percentages of GVdd for the internal VRef. However, it is highly recommended to leave the VREF_OVRD_EN bit cleared to allow VRef to be trained for the best read margins.  000000b - 37% 000001b - 38% 000010b - 39% 111110b - 99% 111111b - 100%
24 VREF_TRAIN_E N	DRAM VRef Train Enable  In addition to the internal VRef training, the DRAM's VRef may also be trained. This bit can be set to automatically enable DRAM VRef training. This bit should be cleared for DDR3 mode.  0b - DRAM VRef will not be trained. 1b - DRAM VRef will be trained.
25 VREF_DRAM_R ANGE	VRef DRAM Range.  If using DRAM VRef training, this bit will specify whether Range 1 or Range 2 will be trained.  0b - DRAM VRef Range 1 will be used for training. 1b - DRAM VRef Range 2 will be used for training.
26-30 —	Reserved
31 ODT	On-Die Termination  ODT termination value for IOs. This is combined with DDRCDR_1[ODT] to determine the termination value. Below is the termination based on concatenating these two fields.  Note that the order of concatenation is (from left to right) DDRCDR_1[ODT], DDRCDR_2[ODT]  Bits [18:19] of the following values are defined at <a href="#">DDRCDR_1.ODT</a> .  000b - Termination is disabled 001b - 120 Ω (DDR3) 100 Ω (DDR4) 010b - 200 Ω (DDR3) 120 Ω (DDR4) 011b - 75 Ω (DDR3) 80 Ω (DDR4) 100b - Reserved (DDR3) 60 Ω (DDR4) 101b - 60 Ω (DDR3) 40 Ω (DDR4) 110b - Reserved (DDR3) 50 Ω (DDR4) 111b - 46 Ω (DDR3) 30 Ω (DDR4)

## 18.4.57 DDR IP block revision 1 (DDR\_IP\_REV1)

### 18.4.57.1 Offset

Register	Offset
DDR_IP_REV1	BF8h

### 18.4.57.2 Function

The DDR IP block revision 1 register provides read-only fields with the IP block ID, along with major and minor revision information.

### 18.4.57.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	IP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	IP_MJ								IP_MN							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1

### 18.4.57.4 Fields

Field	Function
0-15	IP block ID.
IP_ID	For the DDR controller, this value is 0x0002.
16-23	Major revision.
IP_MJ	This is currently set to 0x05.
24-31	Minor revision.
IP_MN	This is currently set to 8'd1.

## 18.4.58 DDR IP block revision 2 (DDR\_IP\_REV2)

### 18.4.58.1 Offset

Register	Offset
DDR_IP_REV2	BFCh

### 18.4.58.2 Function

The DDR IP block revision 2 register provides read-only fields with the IP block integration and configuration options.

### 18.4.58.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								IP_INT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								IP_CFG							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.58.4 Fields

Field	Function
0-7	Reserved
—	
8-15	IP Block Integration Options.
IP_INT	
16-23	Reserved
—	
24-31	IP Block Configuration Options.
IP_CFG	

### 18.4.59 DDR Memory Test Control Register (DDR\_MTCR)

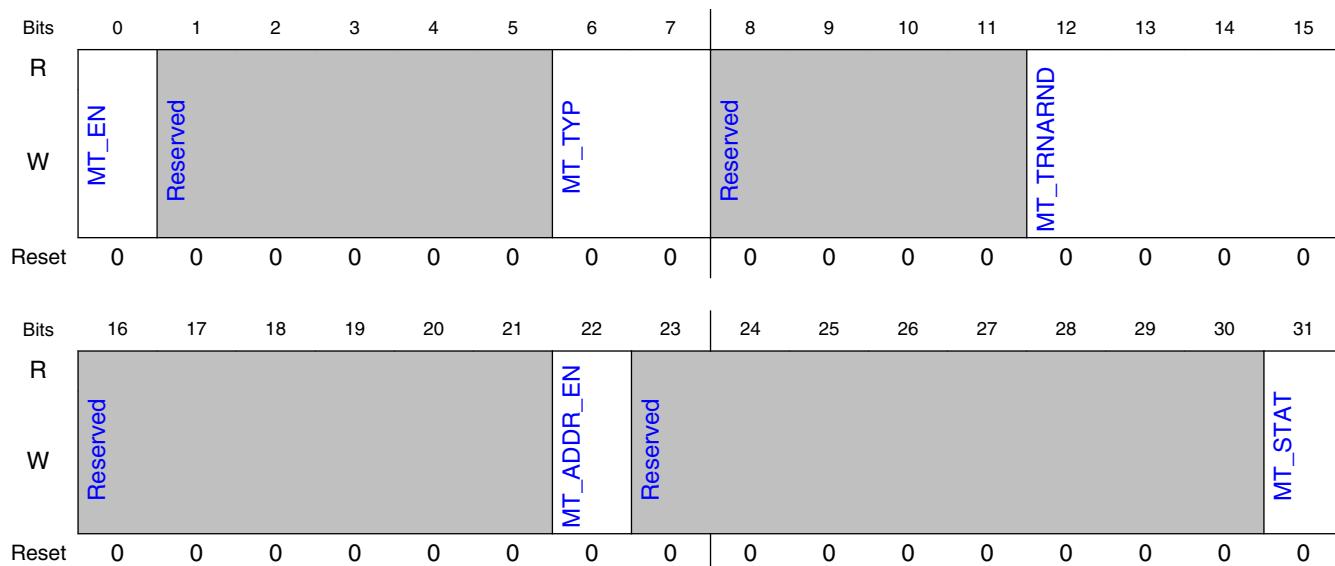
### 18.4.59.1 Offset

Register	Offset
DDR_MTCSR	D00h

### 18.4.59.2 Function

The DDR Memory Test Control Register provides the enable and controls for an automatic memory test.

### 18.4.59.3 Diagram



### 18.4.59.4 Fields

Field	Function
0 MT_EN	Memory Test Enable. This bit can be set by software to enable the memory test. Based on the value of MT_TYP, the controller will either issue writes only, reads only, or it will issue writes and reads. The memory controller will issue transactions throughout all of memory, as defined by the CSn_CONFIG and CSn_BNDS registers. The memory controller will check the data during this test. In addition, the ERR_DETECT register should be read by software if ECC is enabled after the memory test to ensure there were no ECC errors. If an ECC error is detected, the error capture registers will hold the address, data, and attributes captured for the first fail. If there is no ECC error and the test failed (for a data miscompare), then the capture registers will hold information for the transaction that caused the first data miscompare. Note that transactions with

*Table continues on the next page...*

Field	Function
	ECC errors will take priority in the capture registers. Hardware will clear MT_EN after the memory test is complete. This bit can be set before DDR_SDRAM_CFG[MEM_EN] is set, or it can be set again after the memory controller has been enabled.  0b - Memory test has been disabled. 1b - Memory test has been enabled. Hardware will clear this bit when it is complete.
1-5 —	Reserved
6-7 MT_TYP	Memory Test Type.  This field will determine if the memory test will issue writes only, reads only, or both writes and reads. Note that the 'read only' test should not be used unless memory has already been initialized.  00b - Memory test will issue writes and reads. 01b - Memory test will issue writes only. 10b - Memory test will issue reads only. 11b - Reserved
8-11 —	Reserved
12-15 MT_TRNARND	Memory Test Turnaround.  This field determines how many writes will be issued during the memory test before the reads to the same addresses will be issued. This can be used to allow longer streams of writes/reads, and it can be used to test the write->read and read->write turnarounds in a stressful manner. This field is only relevant if MT_TYP is set to 2'b00.  0000b - Entire memory will be written before read transactions are issued. 0001b - Total write/read streams will be 1 transaction each. 0010b - Total write/read streams will be 2 transactions each. 0011b - Total write/read streams will be 4 transactions each. 0100-1111b - Reserved
16-21 —	Reserved
22 MT_ADDR_EN	Memory Test Address Range Enable.  If this bit is set, then the address range defined in the DDR_MT_ST_EXT_ADDR, DDR_MT_ST_ADDR, DDR_MT_END_EXT_ADDR, and DDR_MT_END_ADDR registers will be used.  Please note the following restrictions when using this field to limit the address range to be tested: For any individual test, neither the starting address (DDR_MT_ST_EXT_ADDR    DDR_MT_ST_ADDR) nor the ending address (DDR_MT_END_EXT_ADDR    DDR_MT_END_ADDR) can be less than that used during the previous test. In addition, this field cannot be set if MTCR[MT_TRNARND] is set to 0b0000.  0b - Full memory range defined by CSa_BNDS registers will be used for the memory test. 1b - Memory range defined by DDR_MT_ST_EXT_ADDR, DDR_MT_ST_ADDR, DDR_MT_END_EXT_ADDR, and DDR_MT_END_ADDR is used.
23-30 —	Reserved
31 MT_STAT	Memory Test Status.  After hardware clears MT_EN, this bit will be set if there was a fail. If there is a fail during the memory test (that is, a data miscompare), then this bit will be set at the same time that MT_EN is cleared. Software can clear this bit after it has been set.  0b - No fail has been detected. 1b - A data miscompare was detected during the memory test.

## 18.4.60 DDR Memory Test Pattern n Register (DDR\_MTP0 - DDR\_MTP9)

### 18.4.60.1 Offset

For  $a = 0$  to 9:

Register	Offset
DDR_MTPa	D20h + ( $a \times 4h$ )

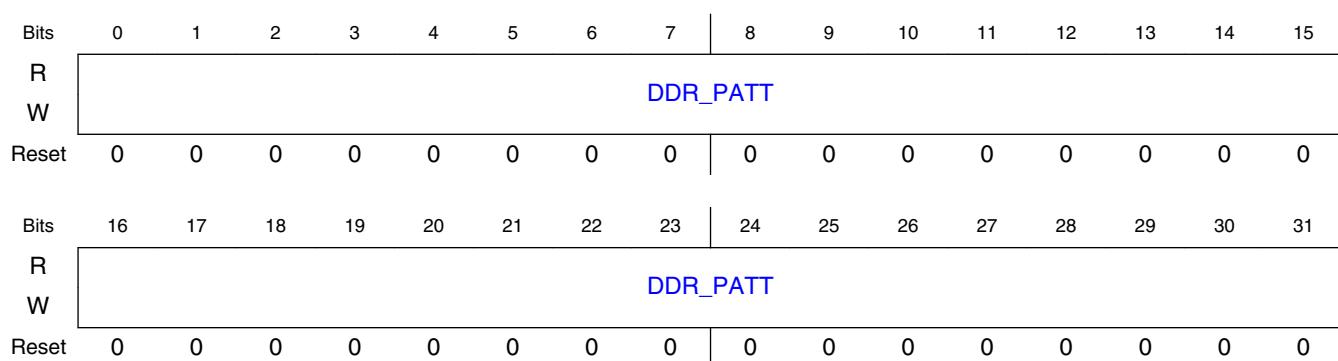
### 18.4.60.2 Function

The DDR memory test pattern  $n$  register provides the data pattern that will be written during the  $n$ th set of 32-bits of each 40-byte memory test pattern. This is used when DDR\_MTCR[MT\_EN] is set to enable the memory write/read test.

#### NOTE

Memory test read compares data that is written in this register.

### 18.4.60.3 Diagram



### 18.4.60.4 Fields

Field	Function
0-31	DDR Pattern.

Field	Function
DDR_PATT	This 32-bit pattern will be used during the memory test if enabled via DDR_MTCR[MT_EN]. This memory test will write/read a programmable 40-byte pattern. The 10 DDR_MTPn registers will create the 40-byte pattern that is used.

## 18.4.61 DDR Memory Test Start Extended Address (DDR\_MT\_ST\_EXT\_ADDR)

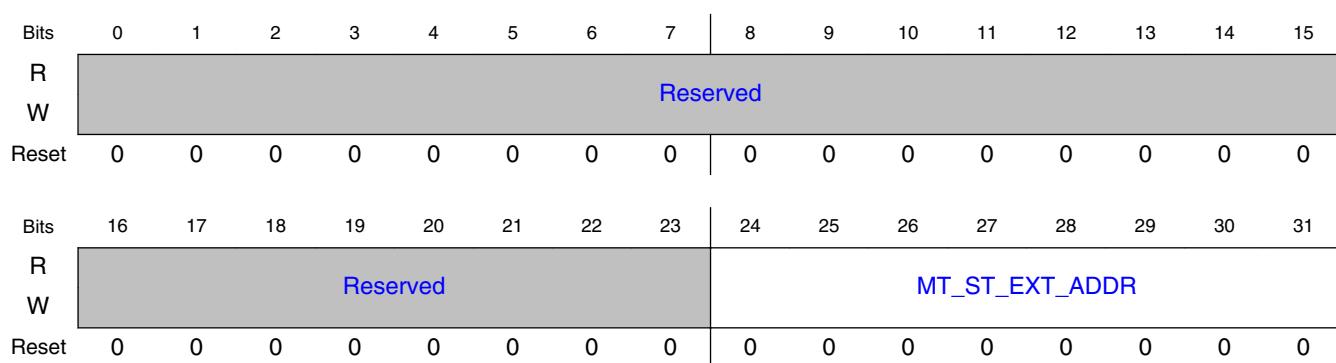
### 18.4.61.1 Offset

Register	Offset
DDR_MT_ST_EXT_ADDR	D60h

### 18.4.61.2 Function

The DDR memory test start extended address register provides the extended address that will be used with DDR\_MTCR[MT\_ADDR\_EN] is set.

### 18.4.61.3 Diagram



## 18.4.61.4 Fields

Field	Function
0-23 —	Reserved
24-31 MT_ST_EXT_A DDR	Memory Test Start Extended Address. This field represents the starting extended address that will be used when MTCR[MT_ADDR_EN] is set. Please note the following restriction when using DDR_MTCR[MT_ADDR_EN] to limit the address range to be tested: For any individual test, neither the starting address (DDR_MT_ST_EXT_ADDR    DDR_MT_ST_ADDR) nor the ending address (DDR_MT_END_EXT_ADDR    DDR_MT_END_ADDR) can be less than that used during the previous test.

## 18.4.62 DDR Memory Test Start Address (DDR\_MT\_ST\_ADDR)

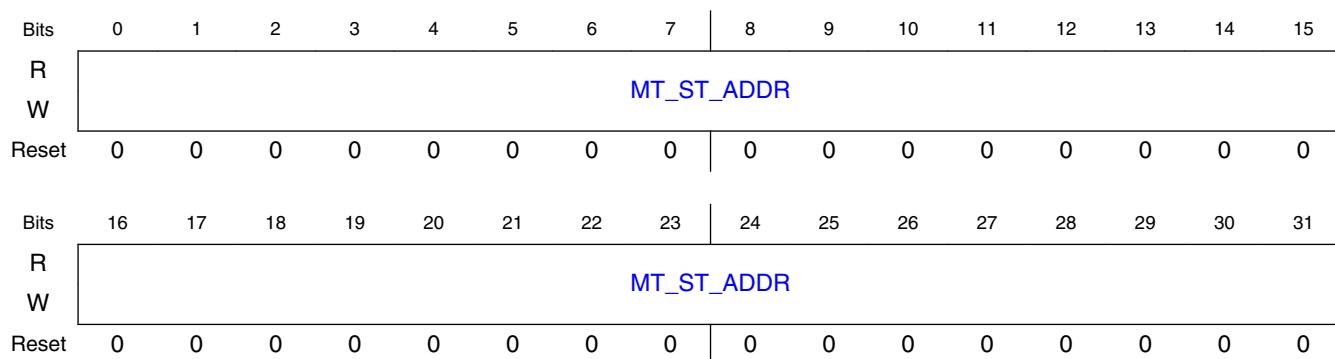
### 18.4.62.1 Offset

Register	Offset
DDR_MT_ST_ADDR	D64h

### 18.4.62.2 Function

The DDR memory test start address register provides the address that will be used with DDR\_MTCR[MT\_ADDR\_EN] is set.

### 18.4.62.3 Diagram



### 18.4.62.4 Fields

Field	Function
0-31 MT_ST_ADDR	Memory Test Start Address.  This field represents the starting address that will be used when MTCR[MT_ADDR_EN] is set.  Please note the following restriction when using DDR_MTCR[MT_ADDR_EN] to limit the address range to be tested: For any individual test, neither the starting address (DDR_MT_ST_EXT_ADDR    DDR_MT_ST_ADDR) nor the ending address (DDR_MT_END_EXT_ADDR    DDR_MT_END_ADDR) can be less than that used during the previous test.

### 18.4.63 DDR Memory Test End Extended Address (DDR\_MT\_E ND\_EXT\_ADDR)

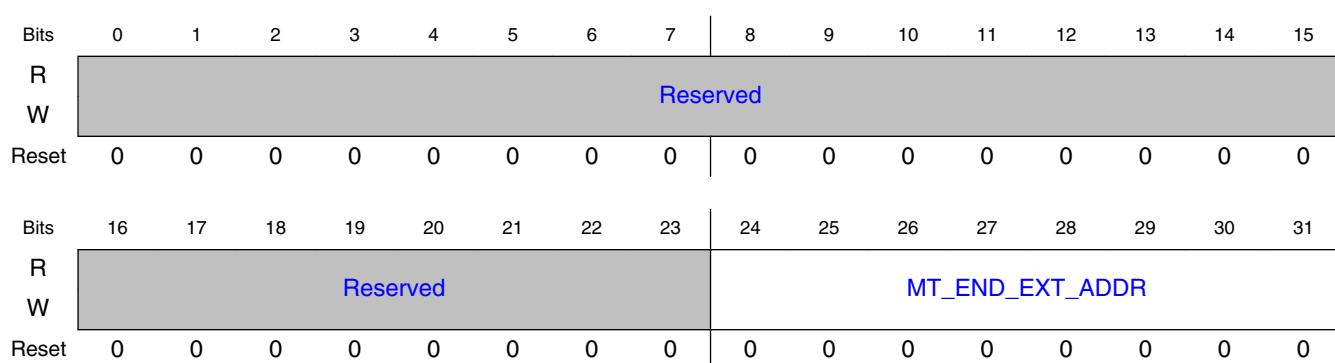
#### 18.4.63.1 Offset

Register	Offset
DDR_MT_END_EXT_A	D68h

#### 18.4.63.2 Function

The DDR memory test end extended address register provides the extended address that will be used with DDR\_MTCR[MT\_ADDR\_EN] is set.

#### 18.4.63.3 Diagram



### 18.4.63.4 Fields

Field	Function
0-23 —	Reserved
24-31 MT_END_EXT_ADDR	Memory Test End Extended Address.  This field represents the ending extended address that will be used when MTCR[MT_ADDR_EN] is set.  Please note the following restriction when using DDR_MTCSR[MT_ADDR_EN] to limit the address range to be tested: For any individual test, neither the starting address (DDR_MT_ST_EXT_ADDR    DDR_MT_ST_ADDR) nor the ending address (DDR_MT_END_EXT_ADDR    DDR_MT_END_ADDR) can be less than that used during the previous test.

### 18.4.64 DDR Memory Test End Address (DDR\_MT\_END\_ADDR)

#### 18.4.64.1 Offset

Register	Offset
DDR_MT_END_ADDR	D6Ch

#### 18.4.64.2 Function

The DDR memory test end address register provides the address that will be used with DDR\_MTCSR[MT\_ADDR\_EN] is set.

### 18.4.64.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									MT_END_ADDR							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									MT_END_ADDR							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.64.4 Fields

Field	Function
0-31 MT_END_ADDR	Memory Test End Address. This field represents the ending address that will be used when MTCR[MT_ADDR_EN] is set. Please note the following restriction when using DDR_MTCR[MT_ADDR_EN] to limit the address range to be tested: For any individual test, neither the starting address (DDR_MT_ST_EXT_ADDR    DDR_MT_ST_ADDR) nor the ending address (DDR_MT_END_EXT_ADDR    DDR_MT_END_ADDR) can be less than that used during the previous test.

## 18.4.65 Memory data path error injection mask high (DATA\_ERR\_INJECT\_HI)

### 18.4.65.1 Offset

Register	Offset
DATA_ERR_INJECT_HI	E00h

## 18.4.65.2 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									EIMH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									EIMH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.65.3 Fields

Field	Function
0-31	Error injection mask high data path.
EIMH	Used to test ECC by forcing errors on the high word of the data path. Setting a bit causes the corresponding data path bit to be inverted on memory bus writes.

## 18.4.66 Memory data path error injection mask low (DATA\_ERR\_INJECT\_LO)

### 18.4.66.1 Offset

Register	Offset
DATA_ERR_INJECT_LO	E04h

## 18.4.66.2 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									EIML							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									EIML							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.66.3 Fields

Field	Function
0-31	Error injection mask low data path.
EIML	Used to test ECC by forcing errors on the low word of the data path. Setting a bit causes the corresponding data path bit to be inverted on memory bus writes.

## 18.4.67 Memory data path error injection mask ECC (ECC\_ERR\_INJECT)

### 18.4.67.1 Offset

Register	Offset
ECC_ERR_INJECT	E08h

### 18.4.67.2 Function

The memory data path error injection mask ECC register sets the ECC mask, enables errors to be written to ECC memory, and allows the ECC byte to mirror the most significant data byte. In addition, a single address parity error may be injected through this register.

### 18.4.67.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	Reserved			Reserved								Reserved			
W																APIEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved						EMB		EIE	N		EEI	M			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.67.4 Fields

Field	Function
0	Reserved
—	
1-3	Reserved
—	
4-11	Reserved
—	
12-14	Reserved
—	
15	Address parity error injection enable. This bit will be cleared by hardware after a single address parity error has been injected. 0b - Address parity error injection disabled. 1b - Address parity error injection enabled.
APIEN	
16-21	Reserved
—	
22	ECC Mirror Byte. 0b - Mirror byte functionality disabled. 1b - Mirror the most significant data path byte onto the ECC byte.
EMB	
23	Error Injection Enable. 0b - Error injection disabled. 1b - Error injection enabled. This applies to the data mask bits, the ECC mask bits, and the ECC mirror bit. Note that error injection should not be enabled until the memory controller has been enabled via DDR_SDRAM_CFG[MEM_EN].
EIEN	
24-31	ECC error injection mask. Setting a mask bit causes the corresponding ECC bit to be inverted on memory bus writes.
EEIM	

## 18.4.68 Memory data path read capture high (CAPTURE\_DATA\_HI)

### 18.4.68.1 Offset

Register	Offset
CAPTURE_DATA_HI	E20h

### 18.4.68.2 Function

The memory data path read capture high register stores the high word of the read data path during error capture.

### 18.4.68.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ECHD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ECHD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.68.4 Fields

Field	Function
0-31	Error capture high data path.
ECHD	Captures the high word of the data path when errors are detected.

## 18.4.69 Memory data path read capture low (CAPTURE\_DATA\_LO)

### 18.4.69.1 Offset

Register	Offset
CAPTURE_DATA_LO	E24h

### 18.4.69.2 Function

The memory data path read capture low register stores the low word of the read data path during error capture.

### 18.4.69.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									ECLD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									ECLD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.69.4 Fields

Field	Function
0-31	Error capture low data path.
ECLD	Captures the low word of the data path when errors are detected.

## 18.4.70 Memory data path read capture ECC (CAPTURE\_ECC)

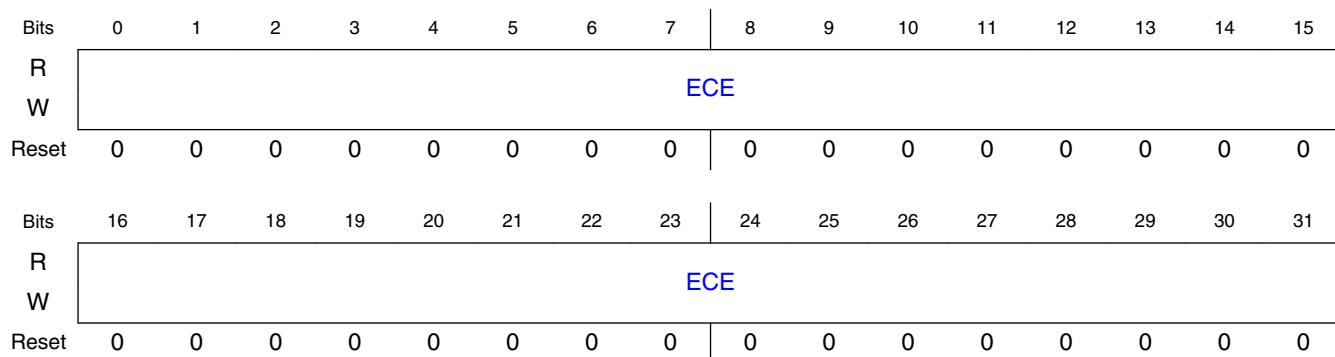
### 18.4.70.1 Offset

Register	Offset
CAPTURE_ECC	E28h

### 18.4.70.2 Function

The memory data path read capture ECC register stores the ECC syndrome bits that were on the data bus when an error was detected.

### 18.4.70.3 Diagram



### 18.4.70.4 Fields

Field	Function
0-31 ECE	Error capture ECC. Captures the ECC bits on the data path whenever errors are detected.  32-bit mode: <ul style="list-style-type: none"><li>0:7 should be ignored</li><li>8:15 8-bit ECC for the 32 bits in beats 0, 2, 4, 6 in 32-bit bus mode</li><li>16:23 should be ignored</li><li>24:31 8-bit ECC for the 32 bits in beats 1, 3, 5, 7 in 32-bit bus mode</li></ul> 16-bit mode: <ul style="list-style-type: none"><li>0:7 8-bit ECC for the 16 bits in beats 0 and 4</li><li>8:15 8-bit ECC for the 16 bits in beats 1 and 5</li><li>16:23 8-bit ECC for the 16 bits in beats 2 and 6</li><li>24:31 8-bit ECC for the 16 bits in beats 3 and 7</li></ul>

## 18.4.71 Memory error detect (ERR\_DETECT)

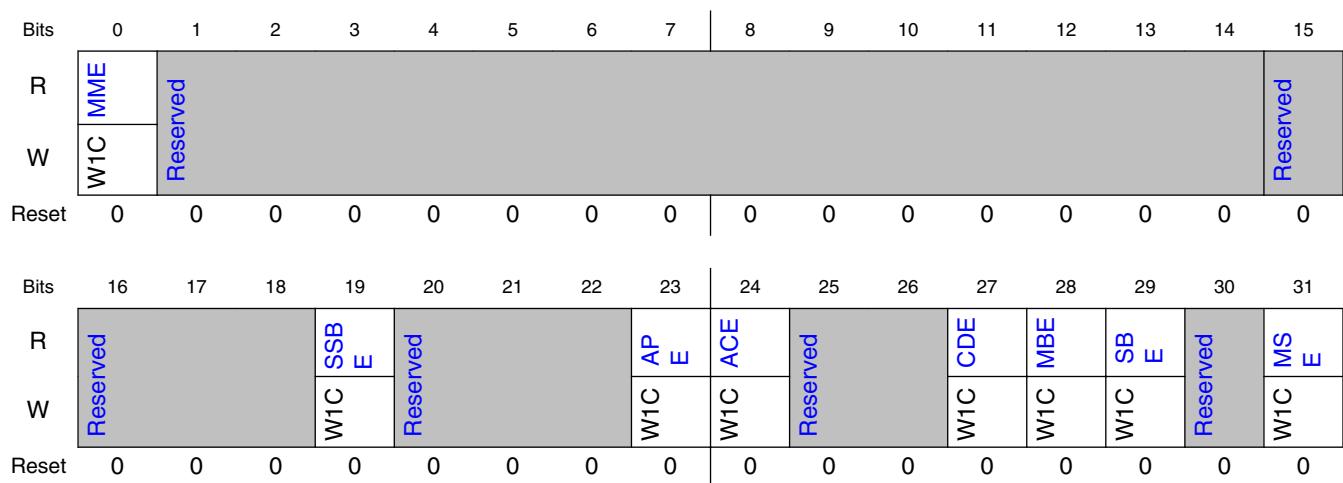
### 18.4.71.1 Offset

Register	Offset
ERR_DETECT	E40h

### 18.4.71.2 Function

The memory error detect register stores the detection bits for multiple memory errors, single- and multiple-bit ECC errors, and memory select errors. It is a read/write register. A bit can be cleared by writing a one to the bit. System software can determine the type of memory error by examining the contents of this register. If an error is disabled with ERR\_DISABLE, the corresponding error is never detected or captured in ERR\_DETECT.

### 18.4.71.3 Diagram



## 18.4.71.4 Fields

Field	Function
0 MME	Multiple memory errors. This bit is cleared by software writing a 1. 0b - Multiple memory errors of the same type were not detected. 1b - Multiple memory errors of the same type were detected.
1-14 —	Reserved
15-18 —	Reserved
19 SSBE	Scrubbed single-bit ECC error. This bit is cleared by software writing a 1. 0b - The number of scrubbed single-bit ECC errors detected has not crossed the threshold set in ERR_SBE[SBET]. 1b - The number of scrubbed single-bit ECC errors detected crossed the threshold set in ERR_SBE[SBET].
20-22 —	Reserved
23 APE	Address parity error. This bit is cleared by software writing a 1. 0b - An address parity error has not been detected. 1b - An address parity error has been detected.
24 ACE	Automatic calibration error. This bit is cleared by software writing a 1. 0b - An automatic calibration error has not been detected. 1b - An automatic calibration error has been detected.
25-26 —	Reserved
27 CDE	Corrupted data error. This bit is cleared by software writing a 1. 0b - A corrupted data error has not been detected. 1b - A corrupted data error has been detected. This bit will be set if the actual ECC is inverted from the expected ECC during a read command. The memory controller will intentionally invert the ECC code if <i>DDR_SDRAM_CFG_2[CD_DIS]</i> is cleared when corrupted data is written to memory. The <i>ERR_DETECT[MBE]</i> bit will also be set when a corrupted data error is detected. Note it is also possible for a 2-bit data error to cause the ECC code to become inverted, which would either mask a corrupted data error or cause a normal multi-bit error to also appear as a corrupted data error.
28 MBE	Multiple-bit error. This bit is cleared by software writing a 1. 0b - A multiple-bit error has not been detected. 1b - A multiple-bit error has been detected.
29 SBE	Single-bit ECC error. This bit is cleared by software writing a 1.

Table continues on the next page...

## DDR register descriptions

Field	Function
	0b - The number of single-bit ECC errors detected has not crossed the threshold set in ERR_SBE[SBET]. 1b - The number of single-bit ECC errors detected crossed the threshold set in ERR_SBE[SBET].
30 —	Reserved
31 MSE	Memory select error. This bit is cleared by software writing a 1. 0b - A memory select error has not been detected. 1b - A memory select error has been detected.

## 18.4.72 Memory error disable (ERR\_DISABLE)

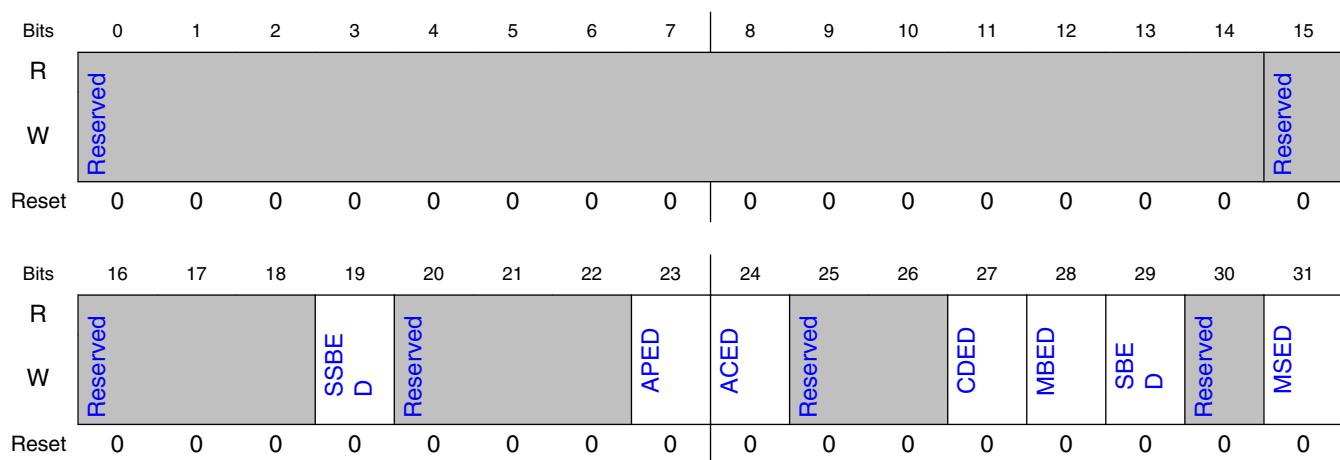
### 18.4.72.1 Offset

Register	Offset
ERR_DISABLE	E44h

### 18.4.72.2 Function

The memory error disable register allows selective disabling of the DDR controller's error detection circuitry. Disabled errors are not detected or reported.

### 18.4.72.3 Diagram



## 18.4.72.4 Fields

Field	Function
0-14 —	Reserved
15-18 —	Reserved
19 SSBED	Scrubbed single-bit ECC error disable.  Note that if this bit is set, then single-bit ECC errors will not be automatically fixed by hardware, even if ECC scrubbing is enabled.  0b - Scrubbed single-bit ECC errors are enabled. 1b - Scrubbed single-bit ECC errors are disabled.
20-22 —	Reserved
23 APED	Address parity error disable. 0b - Address parity errors are detected if DDR_SDRAM_CFG_2[AP_EN] is set. They are reported if ERR_INT_EN[APEE] is set. 1b - Address parity errors are not detected or reported.
24 ACED	Automatic calibration error disable. 0b - Automatic calibration errors are enabled. 1b - Automatic calibration errors are disabled.
25-26 —	Reserved
27 CDED	Corrupted data error disable. 0b - Corrupted data error checking is enabled. 1b - Corrupted data error checking is disabled.
28 MBED	Multiple-bit ECC error disable. 0b - Multiple-bit ECC errors are detected if DDR_SDRAM_CFG[ECC_EN] is set. They are reported if ERR_INT_EN[MBEE] is set. See <a href="#">Error Management</a> for more information. MBED must be zero and ERR_INT_EN[MBEE] and ECC_EN must be one to ensure that an interrupt is generated. 1b - Multiple-bit ECC errors are not detected or reported.
29 SBED	Single-bit ECC error disable.  Note that if this bit is set, then single-bit ECC errors will not be automatically fixed by hardware, even if ECC fixing is enabled.  0b - Single-bit ECC errors are enabled. 1b - Single-bit ECC errors are disabled.
30 —	Reserved
31 MSED	Memory select error disable. 0b - Memory select errors are enabled. 1b - Memory select errors are disabled.

## 18.4.73 Memory error interrupt enable (ERR\_INT\_EN)

### 18.4.73.1 Offset

Register	Offset
ERR_INT_EN	E48h

### 18.4.73.2 Function

The memory error interrupt enable register enables ECC interrupts or memory select error interrupts. When an enabled interrupt condition occurs, the internal int\_B signal is asserted to the programmable interrupt controller (PIC).

### 18.4.73.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W																Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			SSBE E	Reserved			APE E	ACE E	Reserved		CDE E	MBE E	SBE E	Reserved	MSE E
W				0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 18.4.73.4 Fields

Field	Function
0-14	Reserved
—	
15-18	Reserved
—	
19	Scrubbed single-bit ECC error interrupt enable.

Table continues on the next page...

Field	Function
SSBEE	0b - Scrubbed single-bit ECC errors cannot generate interrupts. 1b - Scrubbed single-bit ECC errors generate interrupts.
20-22 —	Reserved
23 APEE	Address parity error interrupt enable. 0b - Address parity errors cannot generate interrupts. 1b - Address parity errors generate interrupts.
24 ACEE	Automatic calibration error interrupt enable. 0b - Automatic calibration errors cannot generate interrupts. 1b - Automatic calibration errors generate interrupts.
25-26 —	Reserved
27 CDEE	Corrupted data error interrupt enable. 0b - Corrupted data errors cannot generate interrupts. 1b - Corrupted data errors generate interrupts.
28 MBEE	Multiple-bit ECC error interrupt enable. See <a href="#">Error Management</a> for more information. Note that uncorrectable read errors may cause an interrupt. ERR_DISABLE[MBED] must be zero and MBEE and DDR_SDRAM_CFG[ECC_EN] must be set to ensure that an interrupt is generated. 0b - Multiple-bit ECC errors cannot generate interrupts. 1b - Multiple-bit ECC errors generate interrupts.
29 SBEE	Single-bit ECC error interrupt enable. 0b - Single-bit ECC errors cannot generate interrupts. 1b - Single-bit ECC errors generate interrupts.
30 —	Reserved
31 MSEE	Memory select error interrupt enable. 0b - Memory select errors do not cause interrupts. 1b - Memory select errors generate interrupts.

## 18.4.74 Memory error attributes capture (CAPTURE\_ATTRIBUTES)

### 18.4.74.1 Offset

Register	Offset
CAPTURE_ATTRIBUTES	E4Ch

## 18.4.74.2 Function

The memory error attributes capture register sets attributes for errors including type, size, source, and others.

## 18.4.74.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	BNUM			Reserved	TSIZ			TSRC							
W									C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	TTYP			Reserved								VLD			
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.74.4 Fields

Field	Function
0 —	Reserved
1-3 BNUM	Data beat number. Captures the doubleword number for the detected error. Relevant only for ECC errors.
4 —	Reserved
5-7 TSIZ	Transaction size for the error. Captures the transaction size in double words.  000b - 8 double words 001b - 1 double word 010b - 2 double words 011b - 3 double words 100b - 4 double words 101b - 5 double word 110b - 6 double words 111b - 7 double words
8-15 TSRC	Transaction source for the error. See the Global Source and Target IDs section in the Memory Map chapter for the defined encoding.

Table continues on the next page...

Field	Function
16-17 —	Reserved
18-19 TTYP	Transaction type for the error. 00b - Reserved 01b - Write 10b - Read 11b - Read-modify-write
20-30 —	Reserved
31 VLD	Valid. Set as soon as valid information is captured in the error capture registers.

## 18.4.75 Memory error address capture (CAPTURE\_ADDRESS)

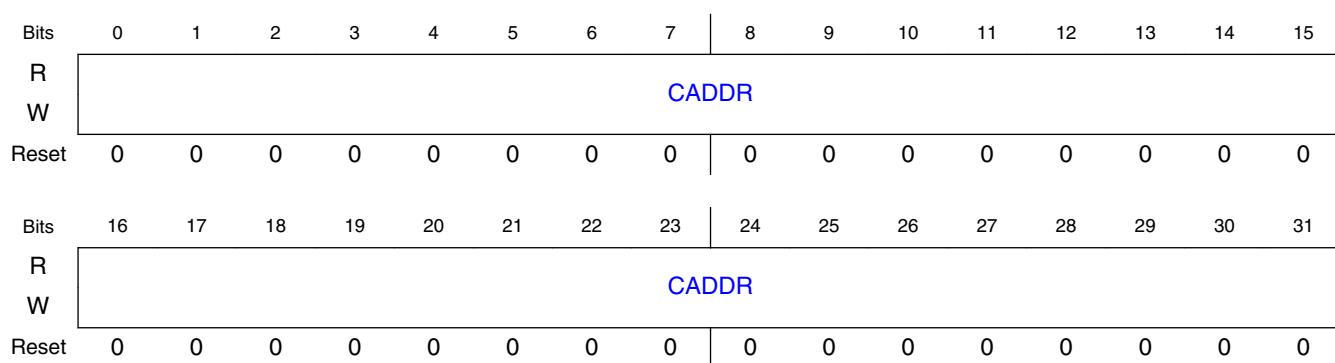
### 18.4.75.1 Offset

Register	Offset
CAPTURE_ADDRESS	E50h

### 18.4.75.2 Function

The memory error address capture register holds the 32 lsbs of a transaction when a DDR ECC error is detected.

### 18.4.75.3 Diagram



### 18.4.75.4 Fields

Field	Function
0-31	Captured address.
CADDR	Captures the 32 lsbs of the transaction address when an error is detected.

### 18.4.76 Memory error extended address capture (CAPTURE\_EXT\_ADDRESS)

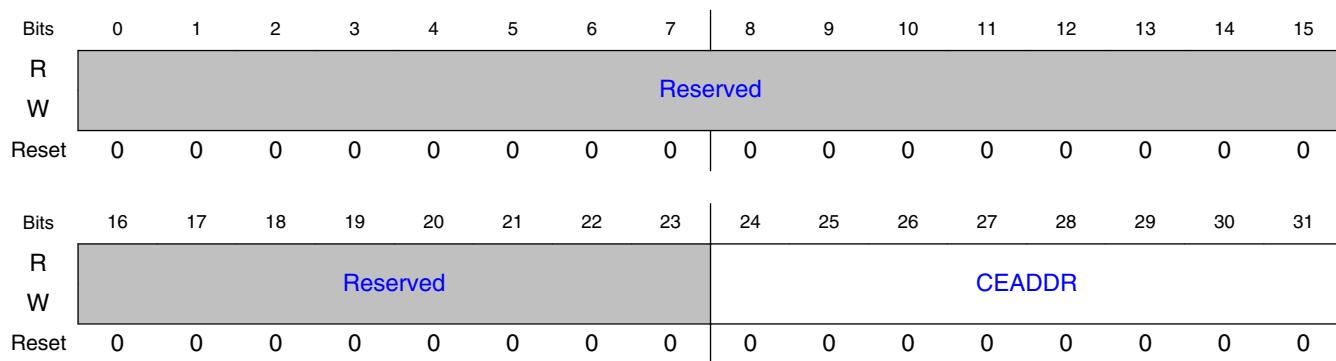
#### 18.4.76.1 Offset

Register	Offset
CAPTURE_EXT_ADDR_ESS	E54h

#### 18.4.76.2 Function

The memory error extended address capture register holds the four most significant transaction bits when an error is detected.

#### 18.4.76.3 Diagram



## 18.4.76.4 Fields

Field	Function
0-23 —	Reserved
24-31 CEADDR	Captured extended address. Captures the 8 msbs of the transaction address when an error is detected

## 18.4.77 Single-Bit ECC memory error management (ERR\_SBE)

### 18.4.77.1 Offset

Register	Offset
ERR_SBE	E58h

### 18.4.77.2 Function

The single-bit ECC memory error management register stores the threshold value for reporting single-bit errors and the number of single-bit errors counted since the last error report. When the counter field reaches the threshold, it wraps back to the reset value (0). If necessary, software must clear the counter after it has managed the error.

### 18.4.77.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SSBET								SBET							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SSBEC								SBEC							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 18.4.77.4 Fields

Field	Function
0-7 SSBET	Scrubbed single-bit error threshold. Establishes the number of single-bit errors that must be detected and fixed during ECC scrubbing before an error condition is reported.
8-15 SBET	Single-bit error threshold. Establishes the number of single-bit errors that must be detected before an error condition is reported.
16-23 SSBEC	Scrubbed single-bit error counter. Indicates the number of scrubbed single-bit errors detected and fixed in memory since the last error report. This counter is incremented only if a single-bit error is detected during the ECC interval scrubbing. If single-bit error reporting is enabled, an error is reported and an interrupt is generated when this value equals SSBET. SSBEC is automatically cleared when the threshold value is reached.
24-31 SBEC	Single-bit error counter. Indicates the number of single-bit errors detected and corrected since the last error report. If single-bit error reporting is enabled, an error is reported and an interrupt is generated when this value equals SBET. SBEC is automatically cleared when the threshold value is reached.

## 18.5 DDR Functional Description

The DDR SDRAM controller controls processor and I/O interactions with system memory. The memory system allows a wide range of memory devices to be mapped to any arbitrary chip select, and support is provided for unbuffered DIMMs.

The figure below is a high-level block diagram of the DDR memory controller. Requests are received from the internal mastering device and the address is decoded to generate the physical bank, logical bank, row, and column addresses. The transaction is compared with values in the row open table to determine if the address maps to an open page. If the transaction does not map to an open page, an active command is issued.

Programmable parameters allow for a variety of memory organizations and timings. Optional error checking and correcting (ECC) protection is provided for the DDR SDRAM data bus. Using ECC, the DDR memory controller detects and corrects all single-bit errors within the data bus, detects all double-bit errors within the data bus, and detects all errors within a nibble. The controller allows as many as 64 pages to be open simultaneously. The amount of time (in clock cycles) the pages remain open is programmable with DDR\_SDRAM\_INTERVAL[BSTOPRE].

### NOTE

When ECC DRAM is present ECC should be enabled.

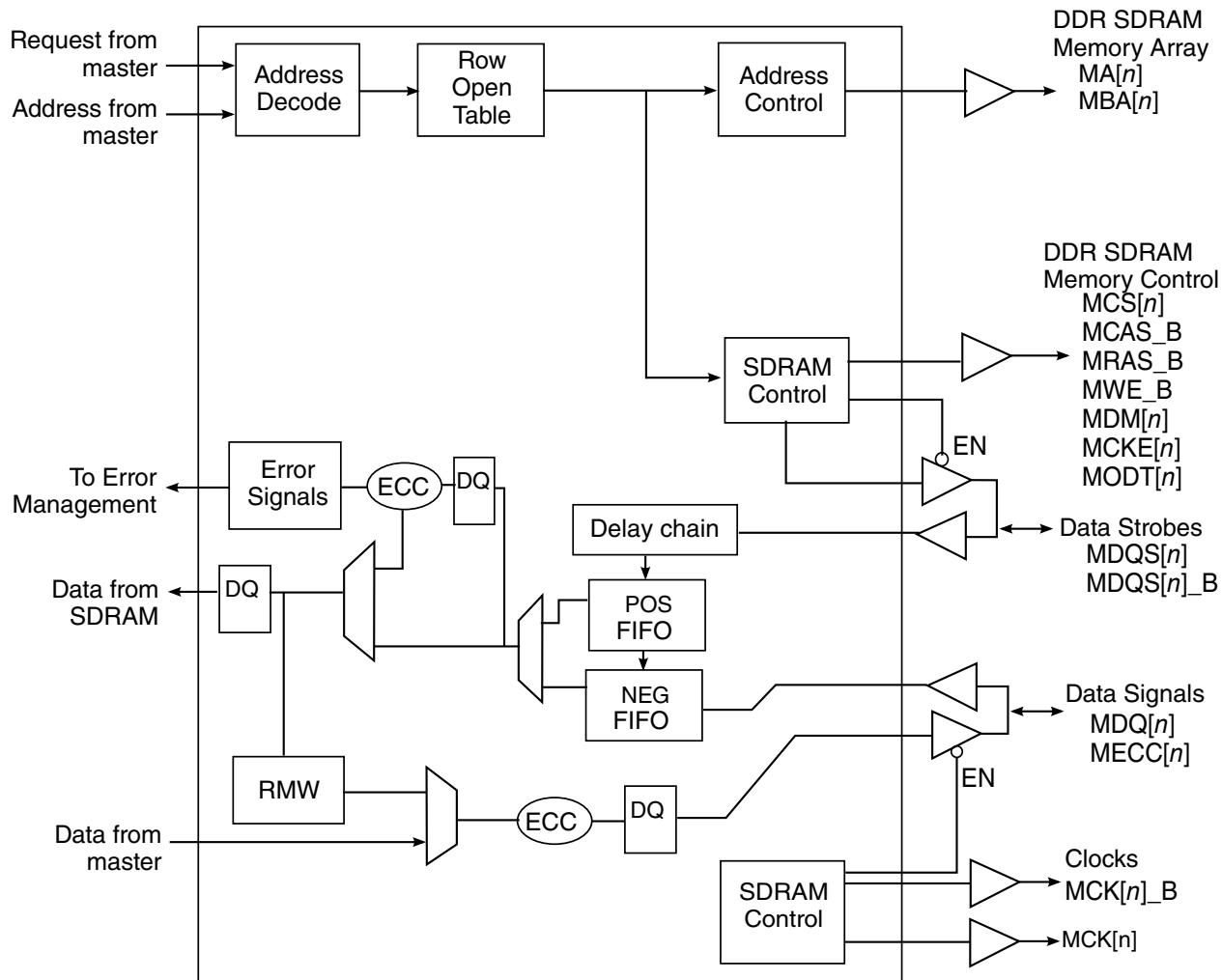


Figure 18-2. DDR Memory Controller Block Diagram

Read and write accesses to memory are burst oriented; accesses start at a selected location and continue for a programmed number of higher locations (4 or 8) in a programmed sequence. Accesses to closed pages start with the registration of an ACTIVE command followed by a READ or WRITE. (Accessing open pages does not require an ACTIVE command.) The address bits registered coincident with the activate command specifies the logical bank and row to be accessed. The address coincident with the READ or WRITE command specify the logical bank and starting column for the burst access.

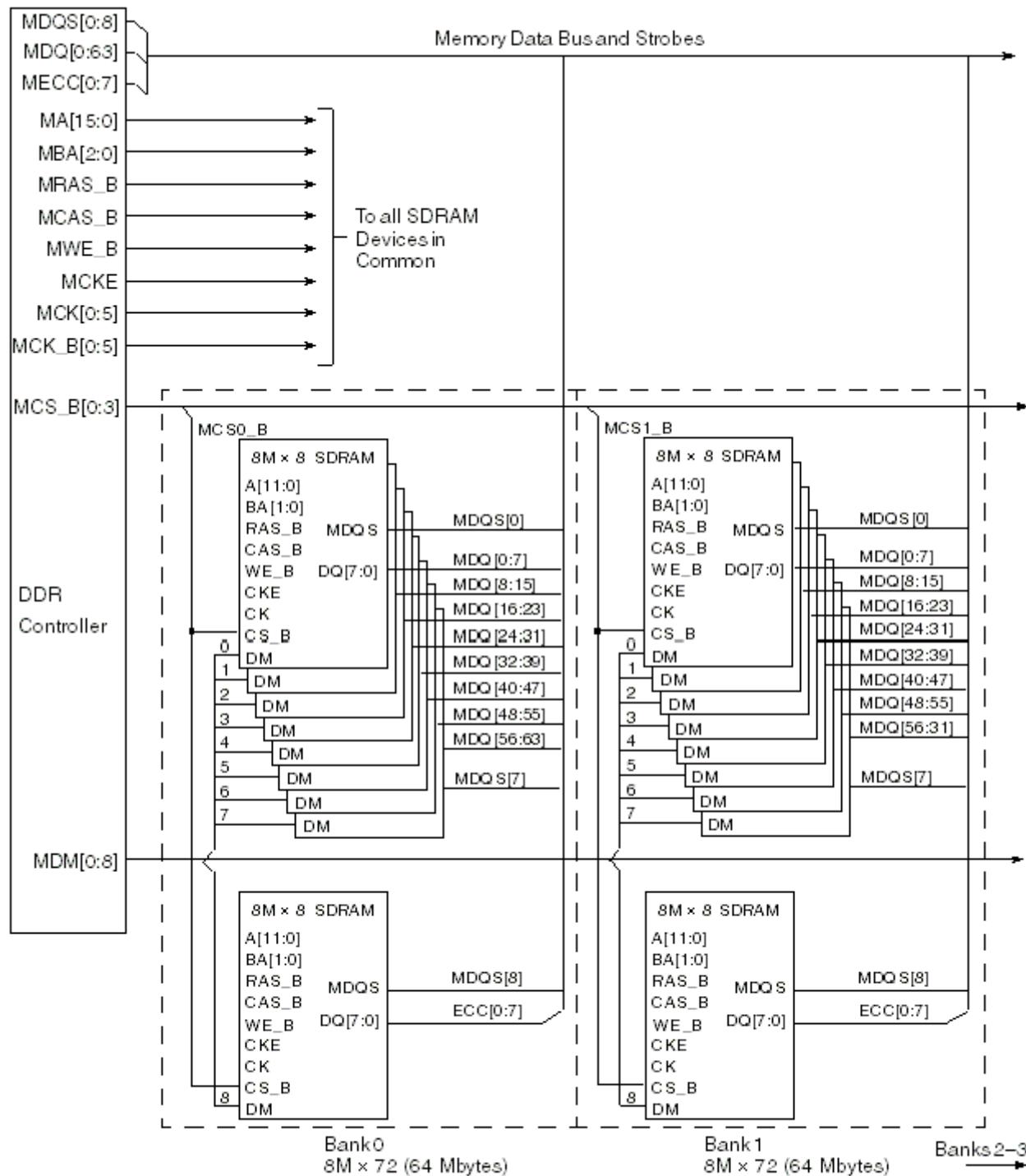
The data interface is source synchronous, meaning whatever sources the data also provides a clocking signal to synchronize data reception. These bidirectional data strobes (MDQS[0:3], MDQS8) are inputs to the controller during reads and outputs during writes. The DDR SDRAM specification requires the data strobe signals to be centered within the data tenure during writes and to be offset by the controller to the center of the data tenure during reads. This delay is implemented in the controller for both reads and writes.

## **DDR Functional Description**

When ECC is enabled, 1 clock cycle is added to the read path to check ECC and correct single-bit errors. ECC generation does not add a cycle to the write path.

The address and command interface is also source synchronous, although 1/16 cycle adjustments are provided for adjusting the clock alignment.

This figure shows an example DDR SDRAM configuration with four physical banks each comprised of four 8M x 8 DDR modules for a total of 256 Mbytes of system memory. One of the nine modules is used for the memory's ECC checking function. Certain address and control lines may require buffering. Analysis of the device's AC timing specifications, desired memory operating frequency, capacitive loads, and board routing loads can assist the system designer in deciding signal buffering requirements. The DDR memory controller drives 16 address pins, but in this example the DDR SDRAM devices use only 12 bits.



1. All signals are connected in common (in parallel) except for MCS\_B[0:3], MCK[0:5], MDM[0:8], and the data bus signals.
2. Each of the MCS\_B[0:3] signals corresponds to a separate physical bank of memory.
3. Buffering may be needed if large memory arrays are used.
4. MCK[0:5] may be apportioned among all memory devices. Complementary bus is not shown.

**Figure 18-3. Example 256-Mbyte DDR SDRAM Configuration**

Error Management explains how the DDR memory controller handles errors.

## 18.5.1 DDR SDRAM Interface Operation

The DDR memory controller supports many different DDR SDRAM configurations; see [Supported DDR SDRAM Organizations](#) for details. SDRAMs with different sizes can be used in the same system. Sixteen multiplexed address signals and three logical bank select signals support numerous device densities. Four chip select (CS\_B) signals support up to two DIMMs of memory. The DDR SDRAM physical banks can be built from standard memory modules or directly-attached memory devices. The data path to individual physical banks is 32 bits wide, 36 bits with ECC. The controller supports physical bank sizes of up to 16 Gbytes. The physical banks can be constructed using x 8, x16, or x32 (with 1 DQS per data byte) memory devices. Five data qualifier (DQM) signals provide byte selection for memory accesses.

### NOTE

An 8-bit DDR SDRAM device has a DQM signal and eight data signals (DQ[0:7]). A 16-bit DDR SDRAM device has two DQM signals associated with specific halves of the 16 data signals (DQ[0:7] and DQ[8:15]).

When ECC is enabled, all memory accesses are performed on double-word boundaries (that is, all DQM signals are set simultaneously). However, when ECC is disabled, the memory system uses the DQM signals for byte lane selection when using x8 or x16 devices.

This table shows the relationships between data byte lanes, MDMn, MDQSn, and MDQn when DDR SDRAM memories are used with x8 or x16 devices.

**Table 18-9. Byte Lane to Data Relationship for x8 and x16 devices**

Data Byte Lane	Data Bus Mask	Data Bus Strobe	Data Bus
0 (MSB)	MDM[0]	MDQS[0]/MDQS_B[0]	MDQ[0:7]
1	MDM[1]	MDQS[1]/MDQS_B[1]	MDQ[8:15]
2	MDM[2]	MDQS[2]/MDQS_B[2]	MDQ[16:23]
3 (LSB)	MDM[3]	MDQS[3]/MDQS_B[3]	MDQ[24:31]
8 (ECC)	MDM8	MDQS8/MDQS8_B	MECC[0:3]

### 18.5.1.1 Supported DDR SDRAM Organizations

These tables describe DDR SDRAM device configurations supported by the DDR memory controller.

**Table 18-10. Supported SDRAM Device Configurations, DDR3 Memory Types**

SDRAM Device	Device Configuration	Row x Column x Sub-bank Bits	32-Bit Bank Size	Four Banks of Memory
512 Mbits	64 Mbits x 8	13 x 10 x 3	256 Mbytes	1 Gbyte
512 Mbits	32 Mbits x 16	12 x 10 x 3	128 Mbytes	512 Mbytes
1 Gbits	128 Mbits x 8	14 x 10 x 3	512 Mbytes	2 Gbytes
1 Gbits	64 Mbits x 16	13 x 10 x 3	256 Mbytes	1 Gbyte
2 Gbits	256 Mbits x 8	15 x 10 x 3	1 Gbyte	4 Gbytes
2 Gbits	128 Mbits x 16	14 x 10 x 3	512 Mbytes	2 Gbytes
4 Gbits	512 Mbits x 8	16 x 10 x 3	2 Gbytes	8 Gbytes
4 Gbits	256 Mbits x 16	15 x 10 x 3	1 Gbyte	4 Gbytes

**Table 18-11. Supported SDRAM Device Configurations, DDR4**

SDRAM Device	Device Configuration	Row x Column x Sub-bank x Bank Group Bits	32-Bit Bank Size	Four Banks of Memory
2 Gbits	256 Mbits x 8	14 x 10 x 2 x 2	1 Gbyte	4 Gbytes
2 Gbits	128 Mbits x 16	14 x 10 x 2 x 1	512 Mbytes	2 Gbytes
4 Gbits	512 Mbits x 8	15 x 10 x 2 x 2	2 Gbytes	8 Gbytes
4 Gbits	256 Mbits x 16	15 x 10 x 2 x 1	1 Gbyte	4 Gbytes
8 Gbits	1 Gbit x 8	16 x 10 x 2 x 2	4 Gbytes	16 Gbytes
8 Gbits	512 Mbits x 16	16 x 10 x 2 x 1	2 Gbytes	8 Gbytes
16 Gbits	2 Gbits x 8	17 x 10 x 2 x 2	8 Gbytes	32 Gbytes
16 Gbits	1 Gbit x 16	17 x 10 x 2 x 1	4 Gbytes	16 Gbytes

If a transaction request is issued to the DDR memory controller and the address does not lie within any of the programmed address ranges for an enabled chip select, a memory select error is flagged. Errors are described in detail in [Error Management](#)

By using a memory-polling algorithm at power-on reset or by querying the JEDEC serial presence detect capability of memory modules, system firmware uses the memory-boundary registers to configure the DDR memory controller to map the size of each bank in memory. The memory controller uses its bank map to assert the appropriate `MCSn_B` signal for memory accesses according to the provided bank starting and ending addresses. The memory banks are not required to be mapped to a contiguous address space.

## 18.5.2 DDR SDRAM Address Multiplexing

The following tables show the address bit encodings for each DDR SDRAM configuration. The address presented at the memory controller signals MA[17:0] use MA[17] as the msb and MA[0] as the lsb. Also, MA[10] is used as the auto-precharge bit for reads and writes, so the column address can never use MA[10].

**Table 18-12. Address Multiplexing using DDR3 Memory Types for 32-Bit Data Bus with Interleaving and Partial Array Self Refresh Disabled**

Row x Col	msb	Address from Core Master																												lsb					
		8	9	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3	38-3						
16 x 10 x 3	MRAS_B	1 5	1 4	1 3	1 2	1 1	10	9	8	7	6	5	4	3	2	1	0																		
	MBA																				2	1	0												
	MCAS_B																							9	8	7	6	5	4	3	2	1	0		
15 x 10 x 3	MRAS_B		1 4	1 3	1 2	1 1	10	9	8	7	6	5	4	3	2	1	0																		
	MBA																				2	1	0												
	MCAS_B																							9	8	7	6	5	4	3	2	1	0		
14 x 10 x 3	MRAS_B			1 3	1 2	1 1	10	9	8	7	6	5	4	3	2	1	0																		
	MBA																				2	1	0												
	MCAS_B																							9	8	7	6	5	4	3	2	1	0		
13 x 10 x 3	MRAS_B				1 2	1 1	10	9	8	7	6	5	4	3	2	1	0																		
	MBA																				2	1	0												
	MCAS_B																							9	8	7	6	5	4	3	2	1	0		
12 x 10 x 3	MRAS_B					1 1	10	9	8	7	6	5	4	3	2	1	0																		
	MBA																				2	1	0												
	MCAS_B																							9	8	7	6	5	4	3	2	1	0		

Chip select interleaving is supported for the memory controller, and is programmed in DDR\_SDRAM\_CFG[BA\_INTLV\_CTL]. Interleaving is supported between chip selects 0 and 1 or chip selects 2 and 3. In addition, interleaving between all four chip selects can be enabled. When interleaving is enabled, the chip selects being interleaved must use the

same size of memory. If two chip selects are interleaved, then 1 extra bit in the address decode is used for the interleaving to determine which chip select to access. If four chip selects are interleaved, then two extra bits are required in the address decode.

The following table illustrates examples of address decode when interleaving between two chip selects.

**Table 18-13. Example of Address Multiplexing for 32/64-Bit Data Bus Interleaving Between Two Banks with Partial Array Self Refresh Disabled**

**Table 18-14. Example of Address Multiplexing for 32/64-Bit Data Bus Interleaving Between Two Banks**

*Table continues on the next page...*

**Table 18-14. Example of Address Multiplexing for 32/64-Bit Data Bus Interleaving Between Two Banks (continued)**

		m s b	Address from Core Master																												lsb			
Row x Col		6	7	8	9	1	1	1	1	1	1	1	1	1	1	1	1	1	20	2	2	2	2	2	2	2	3	3	3	3	3	3	37-3	
Row x Col		3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	1	18	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2-0
13 x 10 x 3																				1	0													
MCAS_B																					1	0												
MCAS_B																					9	8	7	6	5	4	3	2	1	0				
MCAS_B																				0														
MCAS_B																				1	0													
MCAS_B																				1	0													
MCAS_B																				9	8	7	6	5	4	3	2	1	0					

Partial Array Self Refresh (PASR) can be enabled for any chip select using the CSn\_CONFIG\_2[PASR\_CFG] fields. If PASR is enabled for a given chip select, then the sub-bank and row decode will be swapped, and the sub-bank will be decoded as the most significant portion of the DRAM address, as shown in this table.

**Table 18-15. Address Multiplexing with Partial Array Self Refresh Enabled**

Row x Col	ms b	Address from Core Master																													lsb					
		8	9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	3	3	3	3	3	3	38-3				
16 x 10 x 3	MRAS_B																		1	1	13	1	1	1	1	9	8	7	6	5	4	3	2	1	0	
MBA																			5	4	1	2	1	0												

Table continues on the next page...

**Table 18-15. Address Multiplexing with Partial Array Self Refresh Enabled (continued)**

Row x Col	ms b	Address from Core Master																														lsb		
		8	9	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3			
12 x 10 x 3																																		38-3
																																	9	
																																	8	
																																	7	
																																	6	
																																	5	
																																	4	
																																	3	
																																	2	
																																	1	
																																	0	
13 x 10 x 3																																	38-3	
																																	9	
																																	8	
																																	7	
																																	6	
																																	5	
																																	4	
																																	3	
																																	2	
																																	1	
																																	0	
14 x 10 x 3																																	38-3	
																																	9	
																																	8	
																																	7	
																																	6	
																																	5	
																																	4	
																																	3	
																																	2	
																																	1	
																																	0	
15 x 10 x 3																																	38-3	
																																	9	
																																	8	
																																	7	
																																	6	
																																	5	
																																	4	
																																	3	
																																	2	
																																	1	
																																	0	

### 18.5.3 DDR SDRAM Write Timing Adjustments

The DDR memory controller facilitates system design flexibility by providing a write timing adjustment calculated by write leveling for data and DQS. The DDR SDRAM specification requires DQS be received no sooner than 75% of an SDRAM clock period-and no later than 125% of a clock period-from the capturing clock edge of the command/address at the SDRAM. The write leveling calibration is used to meet this timing requirement for a variety of system configurations, ranging from a system with one DIMM to a fully populated system with two DIMMs. DDR\_WRLVL\_CNTL is used to set up and enable write leveling.

### 18.5.4 DDR SDRAM Refresh

The DDR memory controller supports auto-refresh and self-refresh. Auto refresh is used during normal operation and is controlled by the DDR\_SDRAM\_INTERVAL[REFINT] value; self-refresh is used only when the DDR memory controller is set to enter a sleep power management state. The REFINT value, which represents the number of memory bus clock cycles between refresh cycles, must allow for possible outstanding transactions to complete before a refresh request is sent to the memory after the REFINT value is reached. If a memory transaction is in progress when the refresh interval is reached, the refresh cycle waits for the transaction to complete. In the worst case, the refresh cycle must wait the number of bus clock cycles required by the longest programmed access. To ensure that the latency caused by a memory transaction does not violate the device refresh period, it is recommended that the programmed value of REFINT be less than that required by the SDRAM.

When a refresh cycle is required, the DDR memory controller does the following:

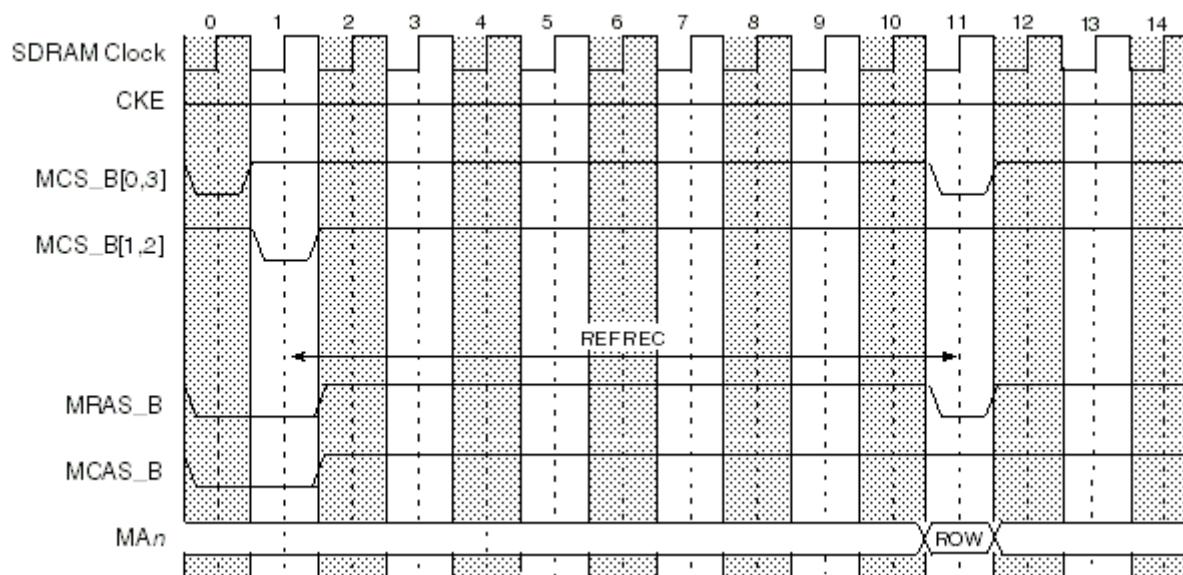
1. Completes all current memory requests.
2. Closes all open pages with a PRECHARGE-ALL command to each DDR SDRAM bank with an open page (as indicated by the row open table).
3. Issues one or more auto-refresh commands to each DDR SDRAM bank (as identified by its chip select) to refresh one row in each logical bank of the selected physical bank.

The auto-refresh commands are staggered across the four possible banks to reduce the system's instantaneous power requirements. Two sets of auto refresh commands will be issued on consecutive cycles when the memory is fully populated with two DIMMs. The initial PRECHARGE-ALL commands are also staggered in two groups for convenience. It is important to note that when entering self-refresh mode, only one refresh command is issued simultaneously to all physical banks. For this entire refresh sequence, no cycle optimization occurs for the usual case where fewer than four banks are installed. After

the refresh sequence completes, any pending memory request is initiated after an inactive period specified by TIMING\_CFG\_1 [REFREC] and TIMING\_CFG\_3[EXT\_REFREC]. In addition, posted refreshes are supported to allow the refresh interval to be set to a larger value.

#### 18.5.4.1 DDR SDRAM Refresh Timing

Refresh timing for the DDR SDRAM is controlled by the programmable timing parameter TIMING\_CFG\_1 [REFREC], which specifies the number of memory bus clock cycles from the refresh command until a logical bank activate command is allowed. The DDR memory controller implements bank staggering for refreshes, as shown in this figure (TIMING\_CFG\_1 [REFREC] = 10 in this example).



**Figure 18-4. DDR SDRAM Bank Staggered Auto Refresh Timing**

System software is responsible for optimal configuration of TIMING\_CFG\_1 [REFREC] and TIMING\_CFG\_3[EXT\_REFREC] at reset. Configuration must be completed before DDR SDRAM accesses are attempted.

#### 18.5.4.2 DDR SDRAM Refresh and Power-Saving Modes

In full-on mode, the DDR memory controller supplies the normal auto refresh to SDRAM. In sleep mode, the DDR memory controller can be configured to take advantage of self-refreshing SDRAMs or to provide no refresh support. Self-refresh support is enabled with the SREN memory control parameter.

This table summarizes the refresh types available in each power-saving mode.

**Table 18-16. DDR SDRAM Power-Saving Modes Refresh Configuration**

Power Saving Mode	Refresh Type	SREN
Sleep	Self	1
	None	-

Note that in the absence of refresh support, system software must preserve DDR SDRAM data (such as by copying the data to disk) before entering the power-saving mode.

The dynamic power-saving mode uses the CKE DDR SDRAM pin to dynamically power down when there is no system memory activity. The CKE pin is negated when both of the following conditions are met:

- No memory refreshes are scheduled
- No memory accesses are scheduled

CKE is reasserted when a new access or refresh is scheduled or the dynamic power mode is disabled. This mode is controlled with DDR\_SDRAM\_CFG[DYN\_PWR\_MGMT].

Dynamic power management mode offers tight control of the memory system's power consumption by trading power for performance through the use of CKE. Powering up the DDR SDRAM when a new memory reference is scheduled causes an access latency penalty, depending on whether active or precharge powerdown is used, along with the settings of TIMING\_CFG\_0[ACT\_PD\_EXIT] and TIMING\_CFG\_0[PRE\_PD\_EXIT].

## 18.5.5 DDR Data Beat Ordering

Transfers to and from memory are always performed in four- or eight-beat bursts. For transfer sizes other than four or eight beats, the data transfers are still operated as four- or eight-beat bursts. If ECC is enabled and either the access is not doubleword aligned or the size is not a multiple of a doubleword, a full read-modify-write is performed for a write to SDRAM. If ECC is disabled or both the access is doubleword aligned with a size that is a multiple of a doubleword, the data masks ((MDM[0:3], MDM8) can be used to prevent the writing of unwanted data to SDRAM. The DDR memory controller also uses data masks to prevent all unintended full double words from writing to SDRAM. For example, if a write transaction is desired with a size of one double word (8 bytes), then the remaining Seven beats of data are not written to DRAM.

All writes for DDR3/DDR4 mode will be aligned to beat 0 of the DRAM.

## 18.5.6 Page Mode and Logical Bank Retention

The DDR memory controller supports an open/closed page mode with an allowable open page for each logical bank (and bank group for DDR4) of DRAM used. In closed page mode for DDR SDRAMs, the DDR memory controller uses the SDRAM auto-precharge feature, which allows the controller to indicate that the page must be automatically closed by the DDR SDRAM after the READ or WRITE access. This is performed by using MA[10] of the address during the COMMAND phase of the access to enable auto-precharge. Auto-precharge is non-persistent in that it is either enabled or disabled for each individual READ or WRITE command. It can, however, be enabled or disabled separately for each chip select.

When the DDR memory controller operates in open page mode, it retains the currently active SDRAM page by not issuing a precharge command. The page remains open until one of the following conditions occurs:

- Refresh interval is met.
- The user-programmable DDR\_SDRAM\_INTERVAL[BSTOPRE] value is exceeded.
- There is a logical bank row collision with another transaction that must be issued.

Page mode can dramatically reduce access latencies for page hits. Depending on the memory system design and timing parameters, using page mode can save two to three clock cycles for subsequent burst accesses that hit in an active page. Also, better performance can be obtained by using more banks, especially in systems which use many different channels. Page mode is disabled by clearing DDR\_SDRAM\_INTERVAL[BSTOPRE] or setting CS n\_CONFIG[AP\_nEN].

## 18.5.7 Error Checking and Correcting (ECC)

The DDR memory controller supports error checking and correcting (ECC) for the data path between the core master and system memory. The memory detects all double-bit errors, detects all multi-bit errors within a nibble, and corrects all single-bit errors. Other errors may be detected, but are not guaranteed to be corrected or detected. Double-bit errors are always reported when error reporting is enabled. When a single-bit error occurs, the single-bit error counter register is incremented, and its value compared to the single-bit error trigger register. An error is reported when these values are equal. The single-bit error registers can be programmed such that minor memory faults are corrected and ignored, but a catastrophic memory failure generates an interrupt.

For writes that are smaller than 64 bits, the DDR memory controller performs a double-word read from system memory of the address for the write (checking for errors), and merges the write data with the data read from memory. Then, a new ECC code is generated for the merged double word. The data and ECC code is then written to

## DDR Functional Description

memory. If a multi-bit error is detected on the read, the transaction completes the read-modify-write to keep the DDR memory controller from hanging. However, the corrupt data is masked on the write, so the original contents in SDRAM remain unchanged.

The DDR controller also supports ECC scrubbing, which is enabled via `DDR_SDRAM_CFG_3[ECC_SCRUB_EN]`. In this mode, all single-bit errors detected will be fixed by hardware. In addition, `DDR_SDRAM_CFG_3[ECC_SCRUB_INT]` can be programmed to enable periodic reads by the DDR controller to search for and fix single-bit errors.

The syndrome encodings for the ECC code are shown in these tables.

**Table 18-17. DDR SDRAM ECC Syndrome Encoding**

Data Bit	Syndrome Bit								Data Bit	Syndrome Bit							
	0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
0	•	•						•	32			•	•				•
1	•		•					•	33			•		•			•
2	•			•				•	34	•		•		•			
3	•				•			•	35		•	•		•			
4	•	•				•			36			•	•		•		
5	•		•			•			37			•		•	•		
6	•			•		•			38	•		•		•	•		•
7	•				•	•			39		•	•		•	•		•
8	•	•					•		40			•	•				•
9	•		•				•		41			•		•			
10	•			•			•		42	•		•		•		•	•
11	•				•		•		43		•	•		•		•	•
12	•	•				•	•	•	44		•	•		•	•	•	•
13	•		•			•	•	•	45			•		•	•	•	•
14	•			•		•	•	•	46	•		•		•	•	•	•
15	•				•	•	•	•	47		•	•		•	•	•	•
16		•	•					•	48		•				•	•	
17		•		•				•	49			•			•	•	
18		•			•			•	50				•		•	•	
19	•	•			•				51	•					•	•	
20		•	•				•		52		•				•		•
21		•		•		•			53			•			•		•
22		•			•	•			54				•		•		•
23	•	•			•	•		•	55	•					•		•
24		•	•					•	56		•				•		•
25		•			•			•	57				•			•	•
26		•			•		•	•	58				•			•	•

*Table continues on the next page...*

**Table 18-17. DDR SDRAM ECC Syndrome Encoding  
(continued)**

Data Bit	Syndrome Bit								Data Bit	Syndrome Bit							
	0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
27	•	•			•		•	•	59	•						•	•
28		•	•			•	•	•	60				•	•		•	
29		•		•		•	•	•	61	•			•	•		•	•
30		•			•	•	•	•	62		•		•	•		•	•
31	•	•			•	•	•		63			•	•	•		•	•

**Table 18-18. DDR SDRAM ECC Syndrome Encoding (Check Bits)**

Check Bit	Syndrome Bit							
	0	1	2	3	4	5	6	7
0	•							
1		•						
2			•					
3				•				
4					•			
5						•		
6							•	
7								•

## 18.5.8 Error Management

The DDR memory controller detects four different kinds of errors: training, single-bit, multi-bit, and memory select errors. The following discussion assumes all the relevant error detection, correction, and reporting functions are enabled as described in [Memory error interrupt enable \(ERR\\_INT\\_EN\)](#), [Memory error disable \(ERR\\_DISABLE\)](#), and [Memory error detect \(ERR\\_DETECT\)](#)

Single-bit errors are counted and reported based on the ERR\_SBE value. When a single-bit error is detected, the DDR memory controller does the following:

- Corrects the data
- Increments the single-bit error counter ERR\_SBE[SBEC]
- Generates a critical interrupt if the counter value ERR\_SBE[SBEC] equals the programmable threshold ERR\_SBE[SBET]
- Completes the transaction normally

If a multi-bit error is detected for a read, the DDR memory controller logs the error and generates the machine check or critical interrupt (if enabled, as described in [Memory error disable \(ERR\\_DISABLE\)](#)). Another error the DDR memory controller detects is a memory select error, which causes the DDR memory controller to log the error and generate a critical interrupt (if enabled, as described in [Memory error detect \(ERR\\_DETECT\)](#)). This error is detected if the address from the memory request does not fall into any of the enabled, programmed chip select address ranges. For all memory select errors, the DDR memory controller does not issue any transactions onto the pins after the first read has returned data strobes. If the DDR memory controller is not using sample points, then a dummy transaction is issued to DDR SDRAM with the first enabled chip select. In this case, the source port on the pins is forced to 0x1F to show the transaction is not real.

[Table 18-19](#) shows the errors with their descriptions. The final error the memory controller detects is the automatic calibration error. This error is set if the memory controller detects an error during its training sequence.

**Table 18-19. Memory Controller Errors**

Category	Error	Descriptions	Action	Detect Register
Notification	Single-bit ECC threshold	The number of ECC errors has reached the threshold specified in the ERR_SBE.	The error is reported via regular or critical interrupt if enabled.	The error control register only logs read versus write, not full type
Access Error	Multi-bit ECC error	A multi-bit ECC error is detected during a read, or read-modify-write memory operation.	The error is reported via machine check or critical interrupt if enabled.	
	Memory select error	Read, or write, address does not fall within the address range of any of the memory banks.		

### 18.5.9 DDR Rapid Clear of Memory

Upon Hard Fail condition, the security monitor drives an interrupt to the DDR controller indicating the memory needs to be cleared, the DDR controller will clear all of memory defined by the CS<sub>n</sub>\_BNDS registers for all enabled chip selects. DDRDSR\_2[RPD\_ST] will be set once the DDR controller begins clearing memory. At this time, writes to the DDR CCSR space are not allowed to modify the register values. Once the rapid clear of memory sequence is complete, writes to the DDR registers may proceed. The DDRDSR\_2[RPD\_EN] bit is provided so software can determine when the rapid clear of memory sequence is complete and can therefore update registers again. Software can clear the RPD\_ST and RPD\_EN bits after the rapid memory clear is complete.

## 18.6 Initialization/Application Information

At system reset or start-up, initialization software (boot code) must set up the programmable parameters in the memory interface configuration registers. See [DDR register descriptions](#) for more detailed descriptions of the configuration registers. These parameters are shown in this table.

**Table 18-20. Memory Interface Configuration Register Initialization Parameters**

Name	Description	Parameter	Section/Page
CS n_BNDS	Chip select memory bounds	SA EA	<a href="#">Chip select a memory bounds (CS0_BNDS - CS3_BNDS)</a>
CS n_CONFIG	Chip select configuration	CS_EN AP_EN ODT_RD_CFG ODT_WR_CFG BA_BITS_CS ROW_BITS_CS BG_BITS_CS COL_BITS_CS	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
TIMING_CFG_3	Extended timing parameters for fields in TIMING_CFG_1	EXT_PRETOACT EXT_ACTTOPRE EXT_ACTTORW EXT_REFREC EXT_CASLAT EXT_WRREC CNTL_ADJ	<a href="#">DDR SDRAM timing configuration 3 (TIMING_CFG_3)</a>
TIMING_CFG_0	Timing configuration	RWT WRT RRT WWT ACT_PD_EXIT PRE_PD_EXIT MRS_CYC	<a href="#">DDR SDRAM timing configuration 0 (TIMING_CFG_0)</a>
TIMING_CFG_1	Timing configuration	PRETOACT ACTTOPRE ACTTORW CASLAT REFREC	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>

*Table continues on the next page...*

**Table 18-20. Memory Interface Configuration Register Initialization Parameters (continued)**

Name	Description	Parameter	Section/Page
		WRREC ACTTOACT WRTORD	
TIMING_CFG_2	Timing configuration	ADD_LAT WR_LAT RD_TO_PRE CKE_PLS FOUR_ACT	DDR SDRAM timing configuration 2 (TIMING_CFG_2)
DDR_SDRAM_CFG	Control configuration	SREN ECC_EN SDRAM_TYPE DYN_PWR BE_8 DBW T2_EN T3_EN BA_INTLV_CTL HSE BI	DDR SDRAM control configuration (DDR_SDRAM_CFG)
DDR_SDRAM_CFG_2	Control configuration	ODT_CFG NUM_PR OBC_CFG AP_EN D_INIT RCW_EN MD_EN	DDR SDRAM control configuration 2 (DDR_SDRAM_CFG_2)
DDR_SDRAM_MODE	Mode configuration	ESDMODE SDMODE	DDR SDRAM mode configuration (DDR_SDRAM_MODE)
DDR_SDRAM_MODE_2	Mode configuration	ESDMODE2 ESDMODE3	DDR SDRAM mode configuration 2 (DDR_SDRAM_MODE_2)
DDR_SDRAM_INTERVAL	Interval configuration	REFINT BSTOPRE	DDR SDRAM interval configuration (DDR_SDRAM_INTERVAL)
DDR_DATA_INIT	Data initialization configuration register	INIT_VALUE	DDR SDRAM data initialization (DDR_DATA_INIT)
DDR_SDRAM_CLK_CNTL	Clock adjust	CLK_ADJUST	DDR SDRAM clock control (DDR_SDRAM_CLK_CNTL)

*Table continues on the next page...*

**Table 18-20. Memory Interface Configuration Register Initialization Parameters (continued)**

Name	Description	Parameter	Section/Page
DDR_INIT_ADDR	Initialization address	INIT_ADDR	DDR training initialization address (DDR_INIT_ADDR)
DDR_INIT_EXT_ADDRESS	Extended initialization address	INIT_EXT_ADDR	DDR training initialization extended address (DDR_INIT_EXT_ADDRESS)
TIMING_CFG_4	Timing configuration	RWT WRT RRT WWT EXT_RWT EXT_WRT EXT_RRT EXT_WWT EXT_REFINT DLL_LOCK	DDR SDRAM timing configuration 4 (TIMING_CFG_4)
TIMING_CFG_5	Timing configuration	RODT_ON RODT_OFF WODT_ON WODT_OFF	DDR SDRAM timing configuration 5 (TIMING_CFG_5)
TIMING_CFG_6	Timing configuration	HS_CASLAT HS_WRLAT HS_WRREC	DDR SDRAM timing configuration 6 (TIMING_CFG_6)
TIMING_CFG_7	Timing configuration	CKE_RST CKSRE CKSRX PAR_LAT CS_TO_CMD	DDR SDRAM timing configuration 7 (TIMING_CFG_7)
TIMING_CFG_8	Timing configuration	RWT_BG WRT_BG RRT_BG WWT_BG ACTTOACT_BG WRTORD_BG PRE_ALL_REC	DDR SDRAM timing configuration 8 (TIMING_CFG_8)
DDR_ZQ_CNTL	ZQ calibration control	ZQ_EN ZQINIT ZQOPER ZQCS	DDR ZQ calibration control (DDR_ZQ_CNTL)

*Table continues on the next page...*

**Table 18-20. Memory Interface Configuration Register Initialization Parameters (continued)**

Name	Description	Parameter	Section/Page
DDR_WRLVL_CNTL	Write leveling control	WRLVL_EN WRLVL_MRД WRLVL_ODTEN WRLVL_DQSEN WRLVL_SMPL WRLVL_WLR WRLVL_START	<a href="#">DDR write leveling control (DDR_WRLVL_CNTL)</a>
DDR_WRLVL_CNTL_2	Write leveling control	WRLVL_START_1 WRLVL_START_2 WRLVL_START_3 WRLVL_START_4	<a href="#">DDR write leveling control 2 (DDR_WRLVL_CNTL_2)</a>
DDR_WRLVL_CNTL_3	Write leveling control	WRLVL_START_5 WRLVL_START_6 WRLVL_START_7 WRLVL_START_8	<a href="#">DDR write leveling control 3 (DDR_WRLVL_CNTL_3)</a>
DDR_SR_CNTR	Self refresh control	SR_IT	<a href="#">DDR Self Refresh Counter (DDR_SR_CNTR)</a>
DDR_SDRAM_RCW_1	Register control words configuration	RCW0 RCW1 RCW2 RCW3 RCW4 RCW5 RCW6 RCW7	<a href="#">DDR Register Control Words 1 (DDR_SDRAM_RCW_1)</a>
DDR_SDRAM_RCW_2	Register control words configuration	RCW8 RCW9 RCW10 RCW11 RCW12 RCW13 RCW14 RCW15	<a href="#">DDR Register Control Words 2 (DDR_SDRAM_RCW_2)</a>
DDR_SDRAM_CFG_3	Control configuration	DDRC_RST ECC_FIX_EN ECC_SCRUB_INT DM_CFG	<a href="#">DDR SDRAM control configuration 3 (DDR_SDRAM_CFG_3)</a>

*Table continues on the next page...*

**Table 18-20. Memory Interface Configuration Register Initialization Parameters (continued)**

Name	Description	Parameter	Section/Page
DDRCDR_1	Driver control	DHC_EN ODT DSO_C_EN DSO_D_EN DSO_CZ DSO_DZ	<a href="#">DDR Control Driver Register 1 (DDRCDR_1)</a>
DDRCDR_2	Driver control	DSO_CLK_EN DSO_CLKZ	<a href="#">DDR Control Driver Register 2 (DDRCDR_2)</a>

## 18.6.1 Programming Summary

Depending on the memory type used, certain fields must be programmed differently.

This table illustrates the differences in certain fields for DDR3 memory types. Note: This table does not list all fields that must be programmed.

**Table 18-21. Programming Summary, DDR3 memory types**

Parameter	Description	Summary	Section
APn_EN	Chip Select $n$ Auto Precharge Enable	Can be used to place chip select $n$ in auto precharge mode	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
ODT_RD_CFG	Chip Select ODT Read Configuration	Can be enabled to assert ODT if desired. This could be set differently depending on system topology. However, systems with only 1 chip select will typically not use ODT when issuing reads to the memory.	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
ODT_WR_CFG	Chip Select ODT Write Configuration	Can be enabled to assert ODT if desired. This could be set differently depending on system topology. However, ODT will typically be set to assert for the chip select that is getting written to (value would be set to 001).	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
PRETOACT	Precharge to Activate Timing	Should be set according to the specifications for the memory used ( $t_{RP}$ )	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CONFIG_1)</a>
ACTTOPRE	Activate to Precharge Timing	Should be set, along with the Extended Activate to Precharge Timing, according to the specifications for the memory used ( $t_{RAS}$ )	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CONFIG_1)</a>

Table continues on the next page...

**Table 18-21. Programming Summary, DDR3 memory types (continued)**

Parameter	Description	Summary	Section
ACTTORW	Activate to Read/Write Timing	Should be set according to the specifications for the memory used ( $t_{RCD}$ )	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
CASLAT	CAS Latency	Should be set, along with the Extended CAS Latency, to the desired CAS latency	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
REFREC	Refresh Recovery	Should be set, along with the Extended Refresh Recovery, to the specifications for the memory used ( $T_{RFC}$ )	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
WRREC	Write Recovery	Should be set according to the specifications for the memory used ( $t_{WR}$ ). If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{WR} + 2$ DRAM cycles.	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
ACTTOACT	Activate A to Activate B	Should be set according to the specifications for the memory used ( $t_{RRD}$ )	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
WRTORD	Write to Read Timing	Should be set according to the specifications for the memory used ( $t_{WTR}$ ) If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{WTR} + 2$ DRAM cycles.	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
ADD_LAT	Additive Latency	Should be set to the desired additive latency. This must be set to a value less than TIMING_CFG_1[ACTTORW]	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
WR_LAT	Write Latency	Should be set to the desired write latency. The minimum WR_LAT that can be used in 1T timing mode is 5 cycles if DDR_RATE=0	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
RD_TO_PRE	Read to Precharge Timing	Should be set according to the specifications for the memory used ( $t_{RTP}$ ). Time between read and precharge for non-zero value of additive latency (AL) is a minimum of AL + $t_{RTP}$ cycles. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{RTP} + 2$ DRAM cycles.	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
CKE_PLS	Minimum CKE Pulse Width	Should be set according to the specifications for the memory used ( $t_{CKE}$ )	DDR SDRAM timing configuration 2 (TIMING_C FG_2)

*Table continues on the next page...*

**Table 18-21. Programming Summary, DDR3 memory types (continued)**

Parameter	Description	Summary	Section
FOUR_ACT	Four Activate Window	Should be set according to the specifications for the memory used ( $t_{FAW}$ ).	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
BE_8	8-beat burst enable	Set to 1 for fixed 8-beat burst mode. Set to 0 for on-the-fly burst chop mode and ensure DDR_SDRAM_CFG_2[OBC_CFG] = 1. If this bit is set to 0, then other requirements in TIMING_CFG_4 will be needed to ensure $t_{CCD}$ is met.	DDR SDRAM control configuration (DDR_SDRA_M_CFG)
T2_EN	2T Timing Enable	In heavily loaded systems, this can be set to 1 to gain extra timing margin on the interface at the cost of address/command bandwidth.	DDR SDRAM control configuration (DDR_SDRA_M_CFG)
ODT_CFG	ODT Configuration	Can be set for termination at the IOs according to system topology. Typically, if ODT is enabled, then the internal IOs should be set up for termination only during reads to DRAM.	DDR SDRAM control configuration 2 (DDR_SDRA_M_CFG_2)
OBC_CFG	On-The-Fly Burst Chop Configuration	Can be set to 1 if on-the-fly burst chop will be used. This feature can only be used if a 64-bit data bus is used.	DDR SDRAM control configuration 2 (DDR_SDRA_M_CFG_2)
RWT	Read-to-write turnaround for same chip select (in TIMING_CFG_4)	This can be used to force a longer read-to-write turnaround time when accessing the same chip select. This is useful for burst chop mode, as there are some timing requirements to the same chip select that still must be met.	DDR SDRAM timing configuration 4 (TIMING_C FG_4)
WRT	Write-to-read turnaround for same chip select (in TIMING_CFG_4)	This could be used to force a certain turnaround time between a write and read to the same chip select. This is useful for burst chop mode. However, it is expected that TIMING_CFG_1[WRTORD] will be programmed appropriately such that TIMING_CFG_4[WRT] can be set to 0000.	DDR SDRAM timing configuration 4 (TIMING_C FG_4)
RRT	Read-to-read turnaround for same chip select (in TIMING_CFG_4)	Should typically be set to 0010 in burst chop mode (on-the-fly or fixed).	DDR SDRAM timing configuration 4 (TIMING_C FG_4)
WWT	Write-to-write turnaround for same chip select (in TIMING_CFG_4)	Should typically be set to 0010 in burst chop mode (on-the-fly or fixed).	DDR SDRAM timing configuration 4 (TIMING_C FG_4)
ZQ_EN	ZQ Calibration Enable	Should be set to 1. The other fields in DDR_ZQ_CNTL should also be programmed appropriately based on the DRAM specifications.	DDR SDRAM timing configuration 7 (TIMING_C FG_7)

*Table continues on the next page...*

**Table 18-21. Programming Summary, DDR3 memory types (continued)**

Parameter	Description	Summary	Section
WRLVL_EN	Write Leveling Enable	Can be set to 1 if write leveling is desired. Otherwise the value used in TIMING_CFG_2[WR_DATA_DELAY] will be used to shift all bytes during writes to DRAM. If write leveling will be used, all other fields in DDR_WRLVL_CNTL should be programmed appropriately based on the DRAM specifications.	<a href="#">DDR write leveling control (DDR_WRLVL_CNTL)</a>
BSTOPRE	Burst To Precharge Interval	Can be set to any value, depending on the application. Auto precharge can be enabled by setting this field to all 0s.	<a href="#">DDR SDRAM interval configuration (DDR_SDRAM_INTERVAL)</a>

This table illustrates the differences in certain fields for DDR4 memory types. Note: This table does not list all fields that must be programmed.

**Table 18-22. Programming Summary, DDR4 memory types**

Parameter	Description	Summary	Section
APn_EN	Chip Select <i>n</i> Auto Precharge Enable	Can be used to place chip select <i>n</i> in auto precharge mode	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
ODT_RD_CFG	Chip Select ODT Read Configuration	Can be enabled to assert ODT if desired. This could be set differently depending on system topology. However, systems with only 1 chip select will typically not use ODT when issuing reads to the memory.	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
ODT_WR_CFG	Chip Select ODT Write Configuration	Can be enabled to assert ODT if desired. This could be set differently depending on system topology. However, ODT will typically be set to assert for the chip select that is getting written to (value would be set to 001).	<a href="#">Chip select a configuration (CS0_CONFIG - CS3_CONFIG)</a>
PRETOACT	Precharge to Activate Timing	Should be set according to the specifications for the memory used ( $t_{RP}$ )	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>
ACTTOPRE	Activate to Precharge Timing	Should be set, along with the Extended Activate to Precharge Timing, according to the specifications for the memory used ( $t_{RAS}$ )	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>
ACTTORW	Activate to Read/Write Timing	Should be set according to the specifications for the memory used ( $t_{RCD}$ )	<a href="#">DDR SDRAM timing configuration 1 (TIMING_CFG_1)</a>

Table continues on the next page...

**Table 18-22. Programming Summary, DDR4 memory types (continued)**

Parameter	Description	Summary	Section
CASLAT	CAS Latency	Should be set, along with the Extended CAS Latency, to the desired CAS latency	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
REFREC	Refresh Recovery	Should be set, along with the Extended Refresh Recovery, to the specifications for the memory used ( $t_{RFC}$ )	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
WRREC	Write Recovery	Should be set according to the specifications for the memory used ( $t_{WR}$ ). If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{WR} + 2$ DRAM cycles.	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
ACTTOACT	Activate A to Activate B	Should be set according to the specifications for the memory used ( $t_{RRD}$ )	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
WRTORD	Write to Read Timing	Should be set according to the specifications for the memory used ( $t_{WTR}$ ). If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{WTR} + 2$ DRAM cycles.	DDR SDRAM timing configuration 1 (TIMING_C FG_1)
ADD_LAT	Additive Latency	Should be set to the desired additive latency. This must be set to a value less than TIMING_CFG_1[ACTTORW]	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
WR_LAT	Write Latency	Should be set to the desired write latency. The minimum WR_LAT that can be used in 1T timing mode is 5 cycles if DDR_RATE=0	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
RD_TO_PRE	Read to Precharge Timing	Should be set according to the specifications for the memory used ( $t_{RTP}$ ). Time between read and precharge for non-zero value of additive latency (AL) is a minimum of AL + $t_{RTP}$ cycles. If DDR_SDRAM_CFG_2[OBC_CFG] is set, then this should be programmed to $t_{RTP} + 2$ DRAM cycles.	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
CKE_PLS	Minimum CKE Pulse Width	Should be set according to the specifications for the memory used ( $t_{CKE}$ )	DDR SDRAM timing configuration 2 (TIMING_C FG_2)
FOUR_ACT	Four Activate Window	Should be set according to the specifications for the memory used ( $t_{FAW}$ ).	DDR SDRAM timing configuration 2 (TIMING_C FG_2)

*Table continues on the next page...*

**Table 18-22. Programming Summary, DDR4 memory types (continued)**

Parameter	Description	Summary	Section
BE_8	8-beat burst enable	Set to 1 for fixed 8-beat burst mode. Set to 0 for on-the-fly burst chop mode and ensure DDR_SDRAM_CFG_2[OBC_CFG] = 1. If this bit is set to 0, then other requirements in TIMING_CFG_4 will be needed to ensure t <sub>CCD</sub> is met.	<a href="#">DDR SDRAM control configuration (DDR_SDRA_M_CFG)</a>
T2_EN	2T Timing Enable	In heavily loaded systems, this can be set to 1 to gain extra timing margin on the interface at the cost of address/command bandwidth.	<a href="#">DDR SDRAM control configuration (DDR_SDRA_M_CFG)</a>
ODT_CFG	ODT Configuration	Can be set for termination at the IOs according to system topology. Typically, if ODT is enabled, then the internal IOs should be set up for termination only during reads to DRAM.	<a href="#">DDR SDRAM control configuration 2 (DDR_SDRA_M_CFG_2)</a>
OBC_CFG	On-The-Fly Burst Chop Configuration	Can be set to 1 if on-the-fly burst chop will be used. This feature can only be used if a 64-bit data bus is used.	<a href="#">DDR SDRAM control configuration 2 (DDR_SDRA_M_CFG_2)</a>
RWT	Read-to-write turnaround for same chip select (in TIMING_CFG_4)	This can be used to force a longer read-to-write turnaround time when accessing the same chip select. This is useful for burst chop mode, as there are some timing requirements to the same chip select that still must be met.	<a href="#">DDR SDRAM timing configuration 4 (TIMING_CFG_4)</a>
WRT	Write-to-read turnaround for same chip select (in TIMING_CFG_4)	This could be used to force a certain turnaround time between a write and read to the same chip select. This is useful for burst chop mode. However, it is expected that TIMING_CFG_1[WRTORD] will be programmed appropriately such that TIMING_CFG_4[WRT] can be set to 0000.	<a href="#">DDR SDRAM timing configuration 4 (TIMING_CFG_4)</a>
RRT	Read-to-read turnaround for same chip select (in TIMING_CFG_4)	Should typically be set to 0010 in burst chop mode (on-the-fly or fixed).	<a href="#">DDR SDRAM timing configuration 4 (TIMING_CFG_4)</a>
WWT	Write-to-write turnaround for same chip select (in TIMING_CFG_4)	Should typically be set to 0010 in burst chop mode (on-the-fly or fixed).	<a href="#">DDR SDRAM timing configuration 4 (TIMING_CFG_4)</a>
ZQ_EN	ZQ Calibration Enable	Should be set to 1. The other fields in DDR_ZQ_CNTL should also be programmed appropriately based on the DRAM specifications.	<a href="#">DDR SDRAM timing configuration 7 (TIMING_CFG_7)</a>
WRLVL_EN	Write Leveling Enable	Should be set to 1 as write leveling is recommended. All other fields in DDR_WRLVL_CNTL, DDR_WRLVL_CNTL_2, and DDR_WRLVL_CNTL_3 should be programmed appropriately based on the DRAM specifications and board layout specifics.	<a href="#">DDR write leveling control (DDR_WRLVL_CNTL)</a>

*Table continues on the next page...*

**Table 18-22. Programming Summary, DDR4 memory types (continued)**

Parameter	Description	Summary	Section
BSTOPRE	Burst To Precharge Interval	Can be set to any value, depending on the application. Auto precharge can be enabled by setting this field to all 0s.	<a href="#">DDR SDRAM interval configuration (DDR_SDRA_M_INTERVAL)</a>

## 18.6.2 DDR SDRAM Initialization Sequence

After configuration of all parameters is complete, system software must set DDR\_SDRAM\_CFG[MEM\_EN] to enable the memory interface. Note that 500  $\mu$ s must elapse after DRAM clocks are stable (after DDRCDR\_1[DHC\_EN] is set, DDR\_SDRAM\_CLK\_CNTL[CLK\_ADJUST] is set, and any chip select is enabled) before MEM\_EN can be set. A delay loop in the initialization code may be necessary if software is enabling the memory controller. If DDR\_SDRAM\_CFG[BI] is not set, the DDR memory controller will conduct an automatic initialization sequence to the memory, which will follow the memory specifications. If the bypass initialization mode is used, then software can initialize the memory through the DDR\_SDRAM\_MD\_CNTL register.

## 18.6.3 Using Forced Self-Refresh Mode to Implement a Battery-Backed RAM System

This section describes the options offered by this device to support battery-backed main memory.

### 18.6.3.1 Software Based Self-Refresh Scheme

The DDR controller also has a software-programmable bit, DDR\_SDRAM\_CFG\_2[FRC\_SR], that immediately puts main memory into self-refresh mode. See [DDR SDRAM control configuration 2 \(DDR\\_SDRAM\\_CFG\\_2\)](#) for a description of this register.

It is expected that a critical interrupt routine triggered by an external voltage sensing device will have time to set this bit.

### 18.6.3.2 Bypassing Re-initialization During Battery-Backed Operation

The DDR controller offers an initialization bypass feature (DDR\_SDRAM\_CFG[BI]), which system designers may use to prevent re-initialization of main memory during system power-on following an abnormal shutdown. See [DDR SDRAM control configuration \(DDR\\_SDRAM\\_CFG\)](#) for information on this bit and [DDR training initialization address \(DDR\\_INIT\\_ADDR\)](#) for a discussion of avoiding possible ECC errors in this mode.

Note that when this mode is used, the controller will wait a user-programmable number of cycles before issuing any command after the assertion of MCKEn; this number is set in TIMING\_CFG\_4[DLL\_LOCK].

# **Chapter 19**

## **Direct Memory Access Multiplexer (DMAMUX)**

### **19.1 The DMAMUX module as implemented on the chip**

This section provides details about how the DMAMUX module is implemented on the chip.

#### **19.1.1 LS1043A DMAMUX module integration**

The following table describes the DMAMUX module integration into this chip:

**Table 19-1. DMAMUX module integration**

Module	Base address
DMAMUX1	2C1_0000
DMAMUX2	2C2_0000

The remainder of this chapter refers to a single DMAMUX module. Notes are included to indicate variations for multiple instantiations.

#### **19.1.2 LS1043A DMAMUX module special consideration**

##### **19.1.2.1 eDMA and DMAMUX**

The device contains one eDMA and two DMAMUX modules.

The eDMA module implements the following parameter settings in the chip:

**Table 19-2. LS1043A eDMA parameter settings**

eDMA parameters	LS1043A parameter value
Stop mode support	Refers to LPM20 low power mode of the chip
Debug mode support	No

### 19.1.2.1.1 eDMA and DMAMUX channel assignment

The table below provides the channel assignments of eDMA and DMAMUX:

**Table 19-3. eDMA and DMAMUX channel assignment**

eDMA channel assignment	Parameter value
15-0	DMAMUX1
31-16	DMAMUX2
63-32	Reserved

### 19.1.2.1.2 DMAMUX settings

The table below provides the DMAMUX settings used in the chip:

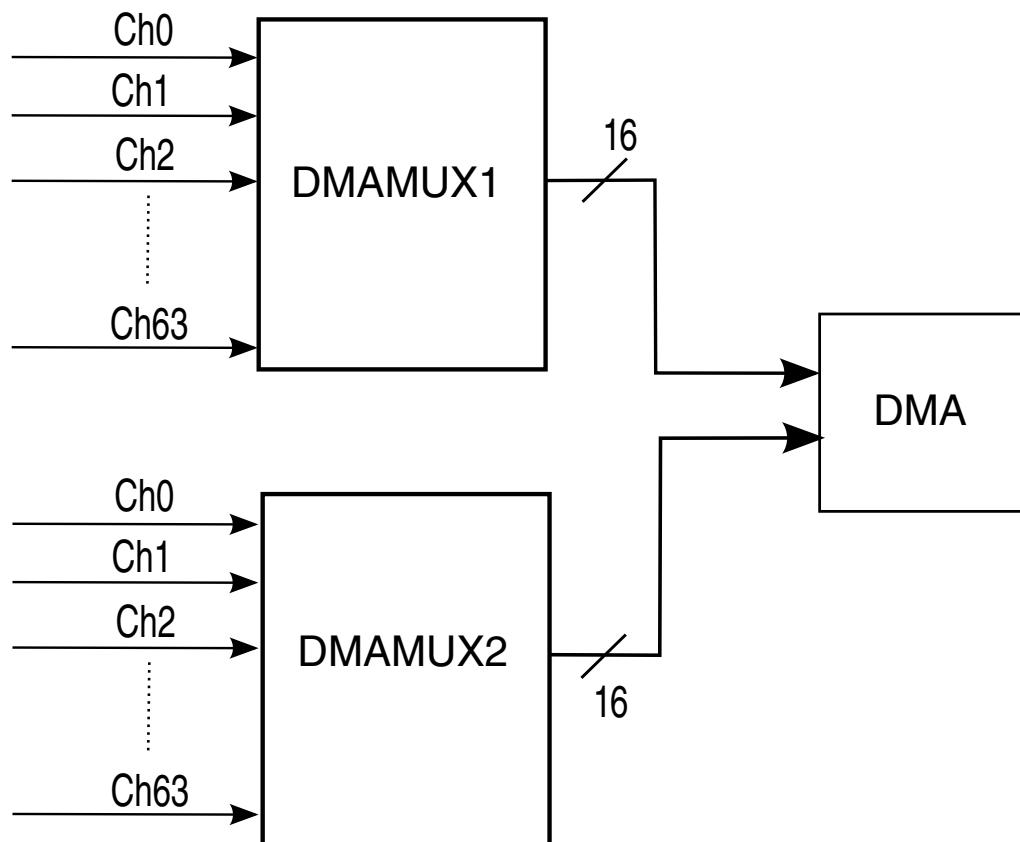
**Table 19-4. DMAMUX settings**

DMAMUX settings	DMAMUX1	DMAMUX2
NUMBER_OF_DMA_CHANNELS	16	16
NUMBER_OF_PERIPHERAL_SLOTS	62	62

### 19.1.2.1.3 DMAMUX request sources

This device includes a DMA request multiplexer that allows up to 64 DMA request signals to be mapped to any of the 16 DMA channels.

As shown in the figure below, request from MUX1 can be mapped to any of the first 16 DMA channels and from MUX2, to any of the lower 16 DMA channels.

**Figure 19-1. DMA Request MUX/DMA Structure**

The table below provides the source mapping for DMAMUX1 and DMAMUX2:

**Table 19-5. DMA-Channel-MUX peripheral mapping**

DMA- Channel- MUX peripheral no.	DMA-CH-MUX1	DMA-CH-MUX2
1	Reserved	Reserved
2	Reserved	Reserved
3	FlexTimer6[0]	Reserved
4	FlexTimer6[1]	Reserved
5	Reserved	Reserved
6	Reserved	Reserved
7	Reserved	Reserved
8	Reserved	Reserved
9	Reserved	Reserved
10	Reserved	Reserved

*Table continues on the next page...*

**Table 19-5. DMA-Channel-MUX peripheral mapping (continued)**

DMA- Channel- MUX peripheral no.	DMA-CH-MUX1	DMA-CH-MUX2
11	FlexTimer5[0]	Reserved
12	FlexTimer5[1]	Reserved
13	Reserved	Reserved
14	Reserved	Reserved
15	Reserved	Reserved
16	Reserved	Reserved
17	Reserved	Reserved
18	Reserved	Reserved
19	FlexTimer4[0]	QSPI rfd
20	FlexTimer4[1]	Reserved
21	FlexTimer4[2]	Reserved
22	FlexTimer4[3]	LPUART6 Rx
23	FlexTimer4[4]	LPUART6 Tx
24	FlexTimer4[5]	LPUART5 Rx
25	FlexTimer4[6]	LPUART5 Tx
26	FlexTimer4[7]	LPUART4 Rx
27	FlexTimer3[0]	LPUART4 Tx
28	FlexTimer3[1]	LPUART3 Rx
29	FlexTimer3[2]	LPUART3 Tx
30	FlexTimer3[3]	LPUART2 Rx
31	FlexTimer3[4]	LPUART2 Tx
32	FlexTimer3[5]	LPUART1 Rx
33	FlexTimer3[6]	LPUART1 Tx
34	FlexTimer3[7]	IIC3 Rx
35	FlexTimer2[0]	IIC3 Tx
36	FlexTimer2[1]	IIC2 Rx
37	FlexTimer2[2]	IIC2 Tx
38	FlexTimer2[3]	IIC1 Rx
39	FlexTimer2[4]	IIC1 Tx
40	FlexTimer2[5]	IIC4 Rx
41	FlexTimer2[6]	IIC4 Tx
42	FlexTimer2[7]	Reserved
43	FlexTimer1[0]	Reserved
44	FlexTimer1[1]	Reserved
45	FlexTimer1[2]	Reserved
46	FlexTimer1[3]	Reserved

*Table continues on the next page...*

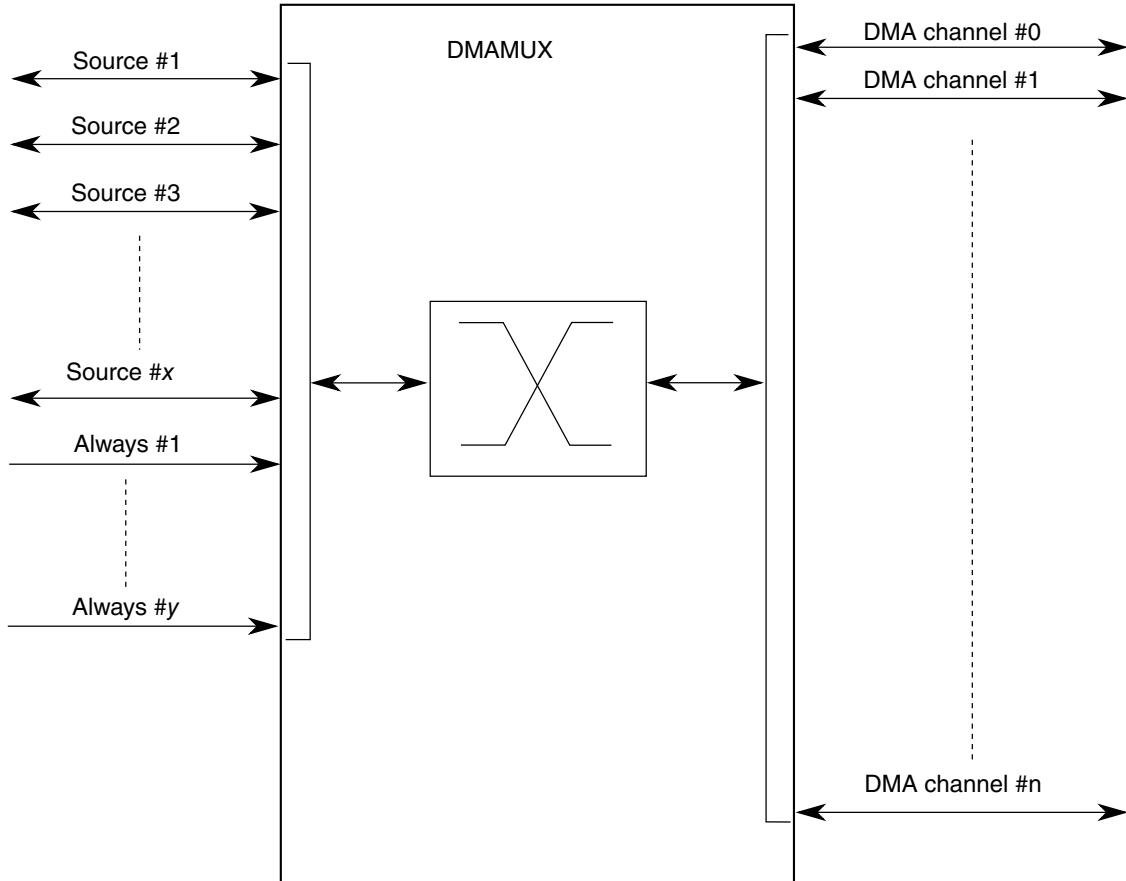
**Table 19-5. DMA-Channel-MUX peripheral mapping (continued)**

DMA- Channel- MUX peripheral no.	DMA-CH-MUX1	DMA-CH-MUX2
47	FlexTimer1[4]	Reserved
48	FlexTimer1[5]	FlexTimer8[0]
49	FlexTimer1[6]	FlexTimer8[1]
50	FlexTimer1[7]	Reserved
51	Reserved	Reserved
52	Reserved	Reserved
53	Reserved	Reserved
54	Reserved	Reserved
55	Reserved	Reserved
56	Reserved	FlexTimer7[0]
57	Reserved	FlexTimer7[1]
58	Reserved	Reserved
59	SPI1 DDIF	Reserved
60	SPI1 RFDF	Reserved
61	SPI1 CMD	Reserved
62	SPI1 TF	Reserved
63	ALWAYS_ENABLED_SOURCE	ALWAYS_ENABLED_SOURCE

## 19.2 Introduction

### 19.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 16 DMA channels. This process is illustrated in the following figure.



**Figure 19-2. DMAMUX block diagram**

## 19.2.2 Features

The DMAMUX module provides these features:

- Up to 62 peripheral slots and up to one always-on slots can be routed to 16 channels.
- 16 independently selectable DMA channel routers.
- Each channel router can be assigned to one of the possible peripheral DMA slots or to one of the always-on slots.

## 19.2.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place.

- Normal mode

In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

## 19.3 External signal description

The DMAMUX has no external pins.

## 19.4 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

**DMAMUX memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C1_0000	Channel Configuration register (DMAMUX1_CHCFG0)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0001	Channel Configuration register (DMAMUX1_CHCFG1)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0002	Channel Configuration register (DMAMUX1_CHCFG2)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0003	Channel Configuration register (DMAMUX1_CHCFG3)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0004	Channel Configuration register (DMAMUX1_CHCFG4)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0005	Channel Configuration register (DMAMUX1_CHCFG5)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0006	Channel Configuration register (DMAMUX1_CHCFG6)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0007	Channel Configuration register (DMAMUX1_CHCFG7)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0008	Channel Configuration register (DMAMUX1_CHCFG8)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_0009	Channel Configuration register (DMAMUX1_CHCFG9)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000A	Channel Configuration register (DMAMUX1_CHCFG10)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000B	Channel Configuration register (DMAMUX1_CHCFG11)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000C	Channel Configuration register (DMAMUX1_CHCFG12)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000D	Channel Configuration register (DMAMUX1_CHCFG13)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000E	Channel Configuration register (DMAMUX1_CHCFG14)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C1_000F	Channel Configuration register (DMAMUX1_CHCFG15)	8	R/W	00h	<a href="#">19.4.1/768</a>

**DMAMUX memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C2_0000	Channel Configuration register (DMAMUX2_CHCFG0)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0001	Channel Configuration register (DMAMUX2_CHCFG1)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0002	Channel Configuration register (DMAMUX2_CHCFG2)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0003	Channel Configuration register (DMAMUX2_CHCFG3)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0004	Channel Configuration register (DMAMUX2_CHCFG4)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0005	Channel Configuration register (DMAMUX2_CHCFG5)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0006	Channel Configuration register (DMAMUX2_CHCFG6)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0007	Channel Configuration register (DMAMUX2_CHCFG7)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0008	Channel Configuration register (DMAMUX2_CHCFG8)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_0009	Channel Configuration register (DMAMUX2_CHCFG9)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000A	Channel Configuration register (DMAMUX2_CHCFG10)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000B	Channel Configuration register (DMAMUX2_CHCFG11)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000C	Channel Configuration register (DMAMUX2_CHCFG12)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000D	Channel Configuration register (DMAMUX2_CHCFG13)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000E	Channel Configuration register (DMAMUX2_CHCFG14)	8	R/W	00h	<a href="#">19.4.1/768</a>
2C2_000F	Channel Configuration register (DMAMUX2_CHCFG15)	8	R/W	00h	<a href="#">19.4.1/768</a>

**19.4.1 Channel Configuration register (DMAMUXx\_CHCFGn)**

Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

**NOTE**

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the source settings, a DMA channel must be disabled via CHCFGn[ENBL].

Address: Base address + 0h offset + (1d × i), where i=0d to 15d

Bit	0	1	2	3	4	5	6	7
Read Write	ENBL	Reserved			SOURCE			
Reset	0	0	0	0	0	0	0	0

**DMAMUX<sub>x</sub>\_CHCFG<sub>n</sub> field descriptions**

Field	Description
0 ENBL	DMA Channel Enable  Enables the DMA channel.  0 DMA channel is disabled. This mode is primarily used during configuration of the DMAMux. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 1 DMA channel is enabled
1 Reserved	This field is reserved. This read/write field does not affect the functionality of the device and should not be used.
2–7 SOURCE	DMA Channel Source (Slot)  Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about the peripherals and their slot numbers.

## 19.5 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels.

As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in [Enabling and configuring sources](#) is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

The DMAMUX channels implement only the normal routing functionality.

### 19.5.1 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are one additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins.
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.

- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source.

## **19.6 Initialization/application information**

This section provides instructions for initializing the DMA channel MUX.

### **19.6.1 Reset**

The reset state of each individual bit is shown in [Memory map/register definition](#). In summary, after reset, all channels are disabled and must be explicitly enabled before use.

## 19.6.2 Enabling and configuring sources

To enable a source, the following steps can be used:

1. Determine with which DMA channel the source will be associated.
2. Clear the CHCFG[ENBL] field of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is set.

To configure source #5 transmit for use with DMA channel 1, as an example, the following steps can be used:

1. Write 0x00 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR      0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;
*CHCFG1 = 0x85;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] bit of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] field is set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8.
3. Write 0x87 to CHCFG8.

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR      0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00;
*CHCFG8 = 0x87;
```

# Chapter 20

## DUART

### 20.1 The DUART module as implemented on the chip

This section provides details about how the DUART module is integrated into this chip.

The chip implements two DUART modules. DUART1 contains UART1 and UART2 and DUART2 contains UART3 and UART4.

**Table 20-1. DUART module as implemented on chip**

Module name	Module base address
DUART1	0x21C_0000
DUART2	0x21D_0000

### 20.2 Overview

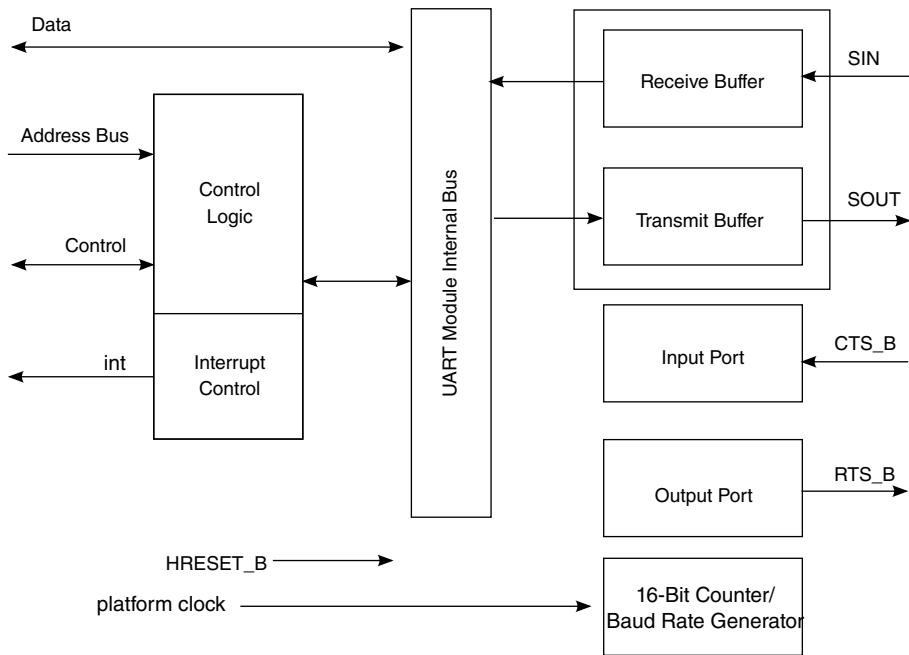
This chapter describes the dual universal asynchronous receiver/transmitters (DUART). It describes the functional operation, the initialization sequence, and the programming details for the dual UART registers and features.

The dual UART consists of two universal asynchronous receiver/transmitters (UARTs). The UARTs act independently; all references to UART refer to one of these receiver/transmitters. Each UART is clocked by the platform clock. The dual UART programming model is compatible with the PC16552D.

The UART interface is point to point, meaning that only two UART devices are attached to the connecting signals. As shown in the figure below, each UART module consists of the following:

- Receive and transmit buffers
- Clear to send (CTS\_B) input port and request to send (RTS\_B) output port for data flow control

- 16-bit counter for baud rate generation
- Interrupt control logic



**Figure 20-1. UART block diagram**

## 20.2.1 Features

The DUART includes these distinctive features:

- Full-duplex operation
- Programming model compatible with original PC16450 UART and PC16550D (improved version of PC16450 that also operates in FIFO mode)
- PC16450 register reset values
- Configurable FIFO mode for both transmitter and receiver, providing 64-byte FIFOs
- Serial data encapsulation and decapsulation with standard asynchronous communication bits (START, STOP, and parity)
- Maskable transmit, receive, line status, and modem status interrupts
- Software-programmable baud generators that divide the platform clock by 1 to  $(2^{16} - 1)$  and generate a 16x clock for the transmitter and receiver engines
- Clear to send (CTS\_B) and ready to send (RTS\_B) modem control functions
- Auto flow for clear to send (CTS\_B) and ready to send (RTS\_B) modem control functions

- Software-selectable serial interface data format (data length, parity, 1/1.5/2 STOP bit, baud rate)
- Line and modem status registers
- Line-break detection and generation
- Internal diagnostic support, local loopback, and break functions
- Prioritized interrupt reporting
- Overrun, parity, and framing error detection

## 20.2.2 Modes of operation

The communication channel provides a full-duplex asynchronous receiver and transmitter using an operating frequency derived from the platform clock .

The transmitter accepts parallel data from a write to the transmitter holding register (UTHR). In FIFO mode, the data is placed directly into an internal transmitter shift register of the transmitter FIFO. The transmitter converts the data to a serial bit stream inserting the appropriate start, stop, and optional parity bits. Finally, it outputs a composite serial data stream on the channel transmitter serial data output signal (SOUT). The transmitter status may be polled or interrupt driven.

The receiver accepts serial data bits on the channel receiver serial data input signal (SIN), converts it to parallel format, checks for a start bit, parity (if any), stop bits, and transfers the assembled character (with start, stop, parity bits removed) from the receiver buffer (or FIFO) in response to a read of the UART's receiver buffer register (URBR). The receiver status may be polled or interrupt driven.

## 20.3 DUART external signal descriptions

The DUART signals are described in the table below.

### NOTE

Although the actual device signal names are prepended with the `UART_` prefix as shown in the table, the abbreviated signal names are often used throughout this chapter.

**Table 20-2. DUART signals-detailed signal descriptions**

Signal	I/O	Description
<code>UARTn_SIN</code>	I	Serial data in. Data is received on the receivers of <code>UARTn</code> through the respective serial data input signal, with the least-significant bit received first. Note that <code>UART3_SIN</code> is not available when <code>UART1</code> is configured for CTS_B mode and <code>UART4_SIN</code> is not available when <code>UART2</code> is configured for CTS_B mode.

*Table continues on the next page...*

**Table 20-2. DUART signals-detailed signal descriptions (continued)**

Signal	I/O	Description	
		<b>State meaning</b>	Asserted/Negated-Represents the data being received on the UART interface.
		<b>Timing</b>	Assertion/Negation-An internal logic sample signal, <i>rxcnt</i> , uses the frequency of the baud-rate generator to sample the data on SIN.
UART $n$ _SOUT	O	Serial data out. The serial data output signals for the UART $n$ are set ('mark' condition) when the transmitter is disabled, operating in the local loopback mode, or idle. Data is shifted out on these signals, with the least significant bit transmitted first. Note that UART3_SOUT is not available when UART1 is configured for RTS_B mode and UART4_SOUT is not available when UART2 is configured for RTS_B mode.	
		<b>State meaning</b>	Asserted/Negated-Represents the data being transmitted on the respective UART interface.
		<b>Timing</b>	Assertion/Negation- An internal logic sample signal, <i>rxcnt</i> , uses the frequency of the baud-rate generator to update and drive the data on SOUT.
UART $n$ _CTS_B	I	Clear to send. These active-low inputs are the clear-to-send inputs. They are connected to the respective RTS_B outputs of the other UART devices on the bus. They can be programmed to generate an interrupt on change-of-state of the signal. Note that UART1_CTS_B is not available when UART3 is configured for SIN mode and UART2_CTS_B is not available when UART4 is configured for SIN mode.	
		<b>State meaning</b>	Asserted/Negated-Represent the clear to send condition for their respective UART.
		<b>Timing</b>	Assertion/Negation-Sampled at the rising edge of every platform clock .
UART $n$ _RTS_B	O	Request to send. UART $n$ _RTS_B are active-low output signals that can be programmed to be negated and asserted by either the receiver or transmitter. When connected to the clear-to-send (CTS_B) input of a transmitter, this signal can be used to control serial data flow. Note that UART1_RTS_B is not available when UART3 is configured for SOUT mode and UART2_RTS_B is not available when UART4 is configured for SOUT mode.	
		<b>State meaning</b>	Asserted/Negated-Represents the data being transmitted on the respective UART interface.
		<b>Timing</b>	Assertion/Negation-Updated and driven at the rising edge of every platform clock .

## 20.4 DUART register descriptions

The table below lists the DUART registers and their offsets. It lists the address, name, and a cross-reference to the complete description of each register.

The UARTs on the device are identical, except that the registers for each UART are located at different offsets. Throughout this chapter, the registers are described by a singular acronym: for example, LCR represents the line control register for each UART (see "The DUART module as implemented on the chip" section for the number of UARTs supported on chip).

The registers in each UART interface are used for configuration, control, and status. The divisor latch access bit, ULCR[DLAB], is used to access the divisor latch least- and most-significant bit registers and the alternate function register. Refer to [UART line control register \(ULCR1 - ULCR2\)](#), for more information on ULCR[DLAB].

All the DUART registers are one-byte wide. Reads and writes to these registers must be byte-wide operations. The table below provides a register summary with references to the section and page that contains detailed information about each register. Undefined byte address spaces within offset 0x000-0xFFFF are reserved.

## 20.4.1 DUART memory map

DUART1 base address: 21C\_0000h

DUART2 base address: 21D\_0000h

Offset	Register	Width (In bits)	Access	Reset value
500h	<a href="#">UART divisor least significant byte register (UDLB1)</a>	8	RW	00h
500h	<a href="#">UART receiver buffer register (URBR1)</a>	8	RO	00h
500h	<a href="#">UART transmitter holding register (UTHR1)</a>	8	WO	00h
501h	<a href="#">UART divisor most significant byte register (UDMB1)</a>	8	RW	00h
501h	<a href="#">UART interrupt enable register (UIER1)</a>	8	RW	00h
502h	<a href="#">UART alternate function register (UAFR1)</a>	8	RW	00h
502h	<a href="#">UART FIFO control register (UFCR1)</a>	8	WO	00h
502h	<a href="#">UART interrupt ID register (UIIR1)</a>	8	RO	01h
503h	<a href="#">UART line control register (ULCR1)</a>	8	RW	00h
504h	<a href="#">UART modem control register (UMCR1)</a>	8	RW	00h
505h	<a href="#">UART line status register (ULSR1)</a>	8	RO	60h
506h	<a href="#">UART modem status register (UMSR1)</a>	8	RO	00h
507h	<a href="#">UART scratch register (USCR1)</a>	8	RW	00h
510h	<a href="#">UART DMA status register (UDSR1)</a>	8	RO	01h
600h	<a href="#">UART divisor least significant byte register (UDLB2)</a>	8	RW	00h
600h	<a href="#">UART receiver buffer register (URBR2)</a>	8	RO	00h
600h	<a href="#">UART transmitter holding register (UTHR2)</a>	8	WO	00h
601h	<a href="#">UART divisor most significant byte register (UDMB2)</a>	8	RW	00h
601h	<a href="#">UART interrupt enable register (UIER2)</a>	8	RW	00h
602h	<a href="#">UART alternate function register (UAFR2)</a>	8	RW	00h
602h	<a href="#">UART FIFO control register (UFCR2)</a>	8	WO	00h
602h	<a href="#">UART interrupt ID register (UIIR2)</a>	8	RO	01h
603h	<a href="#">UART line control register (ULCR2)</a>	8	RW	00h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
604h	UART modem control register (UMCR2)	8	RW	00h
605h	UART line status register (ULSR2)	8	RO	60h
606h	UART modem status register (UMSR2)	8	RO	00h
607h	UART scratch register (USCR2)	8	RW	00h
610h	UART DMA status register (UDSR2)	8	RO	01h

## 20.4.2 UART divisor least significant byte register (UDLB1 - UDLB2)

### 20.4.2.1 Offset

Register	Offset
UDLB1	500h
UDLB2	600h

### 20.4.2.2 Function

This register is accessible when ULCR[DLAB] = 1.

The divisor least significant byte register (UDLB) is concatenated with the divisor most significant byte register (UDMB) to create the divisor used to divide the input clock into the DUART. The output frequency of the baud generator is 16 times the baud rate; therefore the desired baud rate = platform clock frequency  $\div$  (16  $\times$  [UDMB||UDLB]). Equivalently, [UDMB||UDLB:0b0000] = platform clock frequency  $\div$  desired baud rate. Baud rates that can be generated by specific input clock frequencies are shown in the table below.

The following table shows examples of baud rate generation based on common input clock frequencies. Many other target baud rates are also possible.

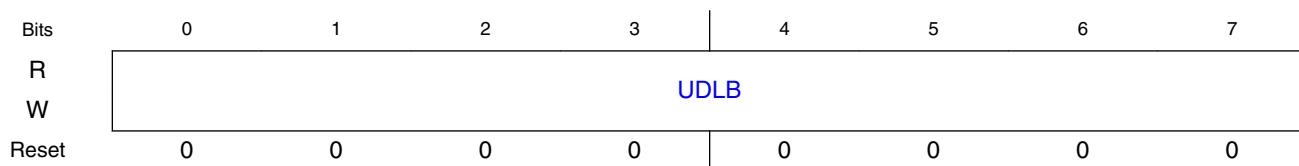
#### NOTE

Because only integer values can be used as divisors, the actual baud rate differs slightly from the desired (target) baud rate; for this reason, both target and actual baud rates are given, along with the percentage of error.

**Table 20-3. Baud Rate Examples**

Target Baud Rate (Decimal)	Divisor		Platform Clock Frequency (MHz)	Actual Baud Rate (Decimal)	Percent Error (Decimal)
	Decimal	Hex			
9,600	1953	07A1	300	9600.61444	0.0064
19,200	977	03D1	300	19,191.40225	0.0448
38,400	488	01E8	300	38,422.13115	0.0576
57,600	326	0146	300	57,515.33742	0.1470
115,200	163	00A3	300	115,030.67485	0.1470
230,400	81	0051	300	231,481.48148	0.4694
9,600	2170	87A	333	9600.61444	0.0064
19,200	1085	43D	333	19,201.22888	0.0064
38,400	543	21F	333	38,367.09638	0.0858
57,600	362	16A	333	57,550.64457	0.0857
115,200	181	B5	333	115,101.28913	0.0857
230,400	90	5A	333	231,481.48148	0.4694
9,600	2604	0A2C	400	9600.61444	0.0064
19,200	1302	0516	400	19,201.22888	0.0064
38,400	651	028B	400	38,402.45776	0.0064
57,600	434	01B2	400	57,603.68664	0.0064
115,200	217	00D9	400	115,207.37327	0.0064
230,400	109	006D	400	229,357.79817	0.4523

### 20.4.2.3 Diagram



### 20.4.2.4 Fields

Field	Function
0-7 UDLB	Divisor least significant byte. This is concatenated with UDMB.

## 20.4.3 UART receiver buffer register (URBR1 - URBR2)

### 20.4.3.1 Offset

Register	Offset
URBR1	500h
URBR2	600h

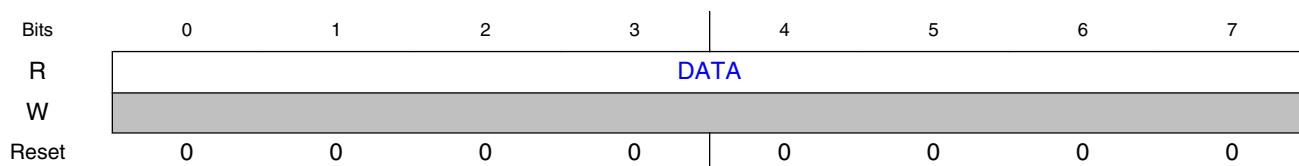
### 20.4.3.2 Function

This register is accessible when ULCR[DLAB] = 0.

These registers contain the data received from the transmitter on the UART buses. In FIFO mode, when read, they return the first byte received. For FIFO status information, refer to the UDSR[RXRDY] description.

Except for the case when there is an overrun, URBR returns the data in the order it was received from the transmitter. Refer to the ULSR[OE] description, [UART line status register \(ULSR1 - ULSR2\)](#). Note that these registers have same offset as the UTHRs.

### 20.4.3.3 Diagram



### 20.4.3.4 Fields

Field	Function
0-7	Data received from transmitter.
DATA	Data received from the transmitter on the UART bus (read only)

## 20.4.4 UART transmitter holding register (UTHR1 - UTHR2)

### 20.4.4.1 Offset

Register	Offset
UTHR1	500h
UTHR2	600h

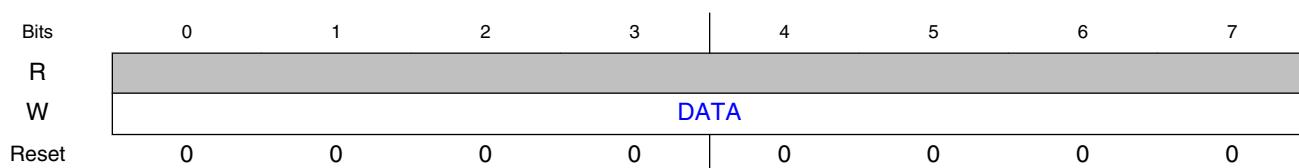
### 20.4.4.2 Function

This register is accessible when ULCR[DLAB] = 0.

A write to these 8-bit registers causes the UART devices to transfer 5-8 data bits on the UART bus in the format set up in the ULCR (line control register). In FIFO mode, data written to UTHR is placed into the FIFO. The data written to UTHR is the data sent onto the UART bus, and the first byte written to UTHR is the first byte onto the bus.

UDSR[TXRDY\_B] indicates when the FIFO is full. See [UART DMA status register \(UDSR1 - UDSR2\)](#) for more details.

### 20.4.4.3 Diagram



### 20.4.4.4 Fields

Field	Function
0-7	Data that is written to UTHR (write only)
DATA	

## 20.4.5 UART divisor most significant byte register (UDMB1 - UDMB2)

### 20.4.5.1 Offset

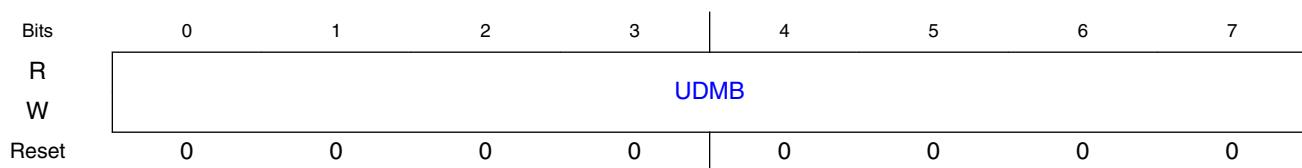
Register	Offset
UDMB1	501h
UDMB2	601h

### 20.4.5.2 Function

This register is accessible when ULCR[DLAB] = 1.

The divisor least significant byte register (UDLB) is concatenated with the divisor most significant byte register (UDMB) to create the divisor used to divide the input clock into the DUART. The output frequency of the baud generator is 16 times the baud rate; therefore the desired baud rate = platform clock frequency  $\div$  (16 x [UDMB||UDLB]). Equivalently, [UDMB||UDLB:0b0000] = platform clock frequency  $\div$  desired baud rate. Baud rates that can be generated by specific input clock frequencies are shown in [UART divisor least significant byte register \(UDLB1 - UDLB2\)](#).

### 20.4.5.3 Diagram



### 20.4.5.4 Fields

Field	Function
0-7 UDMB	Divisor most significant byte

## 20.4.6 UART interrupt enable register (UIER1 - UIER2)

### 20.4.6.1 Offset

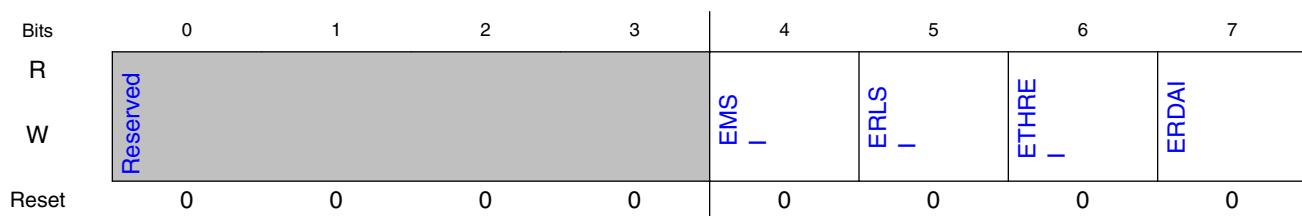
Register	Offset
UIER1	501h
UIER2	601h

### 20.4.6.2 Function

This register is accessible when ULCR[DLAB] = 0.

The UIER gives the user the ability to mask specific UART interrupts to the programmable interrupt controller (PIC).

### 20.4.6.3 Diagram



### 20.4.6.4 Fields

Field	Function
0-3	Reserved.
4 EMSI	Enable modem status interrupt. 0b - Mask interrupts caused by UMSR[DCTS] being set 1b - Enable and assert interrupts when the clear-to-send bit in the UART modem status register (UMSR) changes state
5 ERLSI	Enable receiver line status interrupt. 0b - Mask interrupts when ULSR's overrun, parity error, framing error or break interrupt bits are set

*Table continues on the next page...*

## DUART register descriptions

Field	Function
	1b - Enable and assert interrupts when ULSR's overrun, parity error, framing error or break interrupt bits are set
6 ETHREI	Enable transmitter holding register empty interrupt. 0b - Mask interrupt when ULSR[THRE] is set 1b - Enable and assert interrupts when ULSR[THRE] is set
7 ERDAI	Enable received data available interrupt. 0b - Mask interrupt when new receive data is available or receive data time out has occurred 1b - Enable and assert interrupts when a new data character is received from the external device and/or a time-out interrupt occurs in the FIFO mode

## 20.4.7 UART alternate function register (UAFR1 - UAFR2)

### 20.4.7.1 Offset

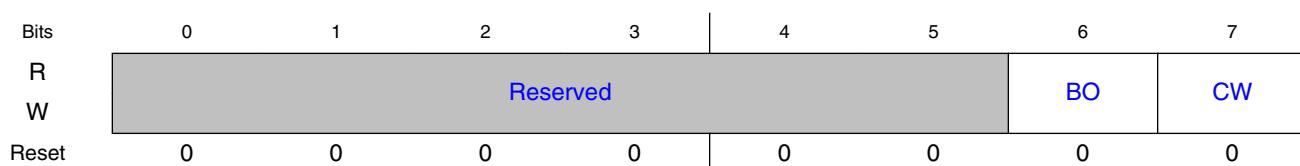
Register	Offset
UAFR1	502h
UAFR2	602h

### 20.4.7.2 Function

This register is accessible when ULCR[DLAB] = 1.

The UAFRs give software the ability to gate off the baud clock and write to each UARTn registers simultaneously with the same write operation.

### 20.4.7.3 Diagram



### 20.4.7.4 Fields

Field	Function
0-5 —	Reserved.
6 BO	Baud clock select. 0b - The baud clock is not gated off. 1b - The baud clock is gated off.
7 CW	Concurrent write enable. 0b - Disables writing to each $\text{UART}_n$ 1b - Enables concurrent writes to corresponding UART registers. A write to a register in $\text{UART}_n$ is also a write to the corresponding register in $\text{UART}_{n+1}$ and vice versa for each DUART where $n$ refers to the number of UART controller. The user needs to ensure that the ULCR[DLAB] of each UART is in the same state before executing a concurrent write to register addresses $0xm00$ , $0xm01$ and $0xm02$ , where $m$ is the base address of the corresponding UART.

### 20.4.8 UART FIFO control register (UFCR1 - UFCR2)

#### 20.4.8.1 Offset

Register	Offset
UFCR1	502h
UFCR2	602h

#### 20.4.8.2 Function

This register is accessible when ULCR[DLAB] = 0.

The UFCR, a write-only register, is used to enable and clear the receiver and transmitter FIFOs, set a receiver FIFO trigger level to control the received data available interrupt, and select the type of DMA signaling.

When the UFCR bits are written, the FIFO enable bit must also be set or else the UFCR bits are not programmed. When changing from FIFO mode to 16450 mode (non-FIFO mode) and vice versa, data is automatically cleared from the FIFOs.

After all the bytes in the receiver FIFO are cleared, the receiver internal shift register is not cleared. Similarly, the bytes are cleared in the transmitter FIFO, but the transmitter internal shift register is not cleared. Both TFR and RFR are self-clearing bits.

### 20.4.8.3 Diagram

Bits	0	1	2	3	4	5	6	7
R								
W	RTL		EN64	Reserved	Reserved	TF R	RF R	FE N
Reset	0	0	0	0	0	0	0	0

### 20.4.8.4 Fields

Field	Function
0-1 RTL	Receiver trigger level. A received data available interrupt occurs when UIER[ERDAI] is set and the number of bytes in the receiver FIFO equals the designated interrupt trigger level as follows:  00b - 1 byte, if EN64 1 byte 01b - 4 bytes, if EN64 16 bytes 10b - 8 bytes, if EN64 32 bytes 11b - 14 bytes, if EN64 56 bytes
2 EN64	Enable 64-byte FIFO  0b - Disables the 64-byte FIFOs 1b - Enables the 64-byte FIFOs
3 —	Reserved
4 —	Reserved
5 TFR	Transmitter FIFO reset  0b - No action 1b - Clears all bytes in the transmitter FIFO and resets the FIFO counter/pointer to 0
6 RFR	Receiver FIFO reset  0b - No action 1b - Clears all bytes in the receiver FIFO and resets the FIFO counter/pointer to 0
7 FEN	FIFO enable  0b - FIFOs are disabled and cleared 1b - Enables the transmitter and receiver FIFOs

### 20.4.9 UART interrupt ID register (UIIR1 - UIIR2)

### 20.4.9.1 Offset

Register	Offset
UIIR1	502h
UIIR2	602h

### 20.4.9.2 Function

This register is accessible when ULCR[DLAB] = 0.

The UIIRs indicate when an interrupt is pending from the corresponding UART and what type of interrupt is active. They also indicate if the FIFOs are enabled.

The DUART prioritizes interrupts into four levels and records these in the corresponding UIIR. The four levels of interrupt conditions in order of priority are:

1. Receiver line status
2. Received data ready/character time-out
3. Transmitter holding register empty
4. Modem status

When the UIIR is read, the associated DUART serial channel freezes all interrupts and indicates the highest priority pending interrupt. While this read transaction is occurring, the associated DUART serial channel records new interrupts, but does not change the contents of UIIR until the read access is complete.

**Table 20-4. UIIR IID Bits Summary**

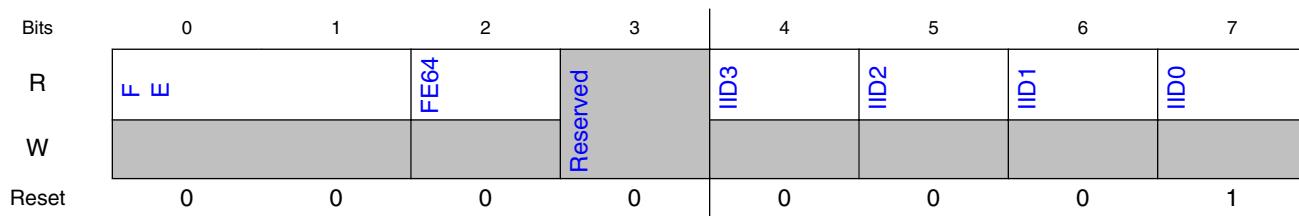
IID Bits IID[3-0]	Priority Level	Interrupt Type	Interrupt Description	How To Reset Interrupt
0b0001	-	-	-	-
0b0110	Highest	Receiver line status	Overrun error, parity error, framing error, or break interrupt	Read the line status register.
0b0100	Second	Received data available	Receiver data available or trigger level reached in FIFO mode	Read the receiver buffer register or interrupt is automatically reset if the number of bytes in the receiver FIFO drops below the trigger level.
0b1100	Second	Character time-out	No characters have been removed from or input to the receiver	Read the receiver buffer register.

*Table continues on the next page...*

**Table 20-4. UIIR IID Bits Summary (continued)**

IID Bits IID[3-0]	Priority Level	Interrupt Type	Interrupt Description	How To Reset Interrupt
			FIFO during the last 4 character times and there is at least one character in the receiver FIFO during this time.	
0b0010	Third	UTHR empty	Transmitter holding register is empty	Read the UIIR or write to the UTHR.
0b0000	Fourth	Modem status	CTS_B input value changed since last read of UMSR	Read the UMSR.

### 20.4.9.3 Diagram



### 20.4.9.4 Fields

Field	Function
0-1 FE	FIFOs enabled. Reflects the setting of UFCR[FEN]
2 FE64	64-byte FIFOs enabled. Reflects the setting of UFCR[EN64]. 0b - 64-byte FIFOs disabled 1b - 64-byte FIFOs enabled
3 —	Reserved
4 IID3	Interrupt ID bits identify the highest priority interrupt that is pending as indicated in the table above. IID3 is set along with IID2 only when a timeout interrupt is pending for FIFO mode.
5 IID2	Interrupt ID bits identify the highest priority interrupt that is pending as indicated in the table above.
6 IID1	Interrupt ID bits identify the highest priority interrupt that is pending as indicated in the table above.

Table continues on the next page...

Field	Function
7 IID0	IID0 indicates when an interrupt is pending. 0b - The UART has an active interrupt ready to be serviced. 1b - No interrupt is pending.

## 20.4.10 UART line control register (ULCR1 - ULCR2)

### 20.4.10.1 Offset

Register	Offset
ULCR1	503h
ULCR2	603h

### 20.4.10.2 Function

This register is accessible when ULCR[DLAB] = x.

The ULCRs specify the data format for the UART bus and set the divisor latch access bit ULCR[DLAB], which controls the ability to access the divisor latch least and most significant bit registers and the alternate function register.

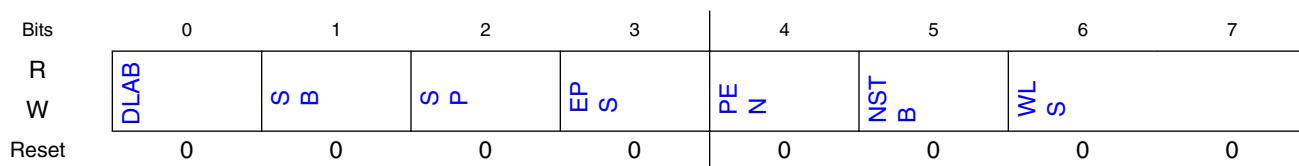
After initializing the ULCR, the software should not re-write the ULCR when valid transfers on the UART bus are active. The software should not re-write the ULCR until the last STOP bit has been received and there are no new characters being transferred on the bus.

The stick parity bit, ULCR[SP], assigns a set parity value for the parity bit time slot sent on the UART bus. The set value is defined as mark parity (logic 1) or space parity (logic 0). ULCR[PEN] and ULCR[EPS] help determine the set parity value. See the table below for more information. ULCR[NSTB], defines the number of STOP bits to be sent at the end of the data transfer. The receiver only checks the first STOP bit, regardless of the number of STOP bits selected. The word length select bits (1 and 0) define the number of data bits that are transmitted or received as a serial character. The word length does not include START, parity, and STOP bits.

**Table 20-5. Parity Selection Using ULCR[PEN], ULCR[SP], and ULCR[EPS]**

PEN	SP	EPS	Parity Selected
0	0	0	No parity
0	0	1	No parity
0	1	0	No parity
0	1	1	No parity
1	0	0	Odd parity
1	0	1	Even parity
1	1	0	Mark parity
1	1	1	Space parity

### 20.4.10.3 Diagram



### 20.4.10.4 Fields

Field	Function
0 DLAB	Divisor latch access bit. 0b - Access to all registers except UDLB, UAFLR, and UDMB 1b - Ability to access divisor latch least and most significant byte registers and alternate function register (UAFLR)
1 SB	Set break. 0b - Send normal UTHR data onto the serial output (SOUT) signal 1b - Force logic 0 to be on the SOUT signal. Data in the UTHR is not affected
2 SP	Stick parity. 0b - Stick parity is disabled. 1b - If PEN = 1 and EPS = 1, space parity is selected. And if PEN = 1 and EPS = 0, mark parity is selected.
3 EPS	Even parity select. See the table above for more information. 0b - If PEN = 1 and SP = 0, odd parity is selected. 1b - If PEN = 1 and SP = 0, even parity is selected.
4 PEN	Parity enable. 0b - No parity generation and checking 1b - Generate parity bit as a transmitter, and check parity as a receiver

Table continues on the next page...

Field	Function
5 NSTB	Number of STOP bits. 0b - One STOP bit is generated in the transmitted data. 1b - When a 5-bit data length is selected, 1½ STOP bits are generated. When either a 6-, 7-, or 8-bit word length is selected, two STOP bits are generated.
6-7 WLS	Word length select. Number of bits that comprise the character length. The word length select values are as follows: 00b - 5 bits 01b - 6 bits 10b - 7 bits 11b - 8 bits

## 20.4.11 UART modem control register (UMCR1 - UMCR2)

### 20.4.11.1 Offset

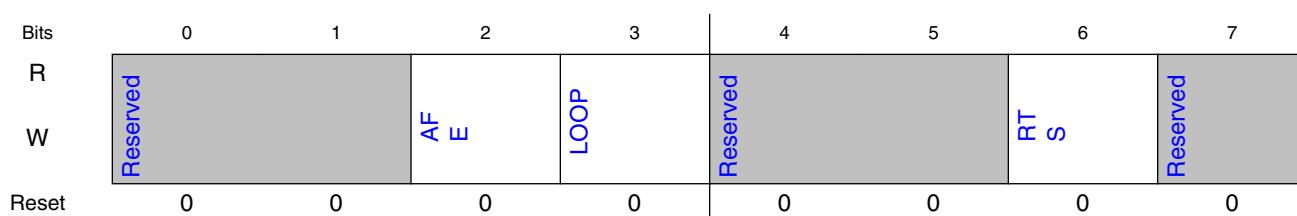
Register	Offset
UMCR1	504h
UMCR2	604h

### 20.4.11.2 Function

This register is accessible when ULCR[DLAB] = x.

The UMCRs control the interface with the external peripheral device on the UART bus.

### 20.4.11.3 Diagram



## 20.4.11.4 Fields

Field	Function
0-1 —	Reserved.
2 AFE	Auto Flow Control Enable Setting this bit to 1 enables the UART's autflow. RTS AFE Auto-flow configuration: 0b - When RTS is either 0 or 1, both Auto-RTS and Auto-CTS are disabled. 1b - When RTS is 0, only Auto-CTS is enabled. When RTS is 1, both Auto-CTS and Auto-RTS are enabled.
3 LOOP	Local loopback mode. 0b - Normal operation 1b - Functionally, the data written to UTHR can be read from URBR of the same UART , and UMCR[RTS] is tied to UMSR[CTS] .
4-5 —	Reserved.
6 RTS	Ready to send. 0b - Negates corresponding RTS_B output 1b - Assert corresponding RTS_B output. Informs external modem or peripheral that the UART is ready for sending/receiving data
7 —	Reserved.

## 20.4.12 UART line status register (ULSR1 - ULSR2)

### 20.4.12.1 Offset

Register	Offset
ULSR1	505h
ULSR2	605h

### 20.4.12.2 Function

This register is accessible when ULCR[DLAB] = x.

The ULSRs are read-only registers that monitor the status of the data transfer on the UART buses. To isolate the status bits from the proper character received through the UART bus, software should read the ULSR and then the URBR.

### 20.4.12.3 Diagram

Bits	0	1	2	3	4	5	6	7
R	RF E	TEM T	THR E	BI	FE E	PE	OE	DR
W								
Reset	0	1	1	0	0	0	0	0

### 20.4.12.4 Fields

Field	Function
0 RFE	Receiver FIFO error. 0b - This bit is cleared when there are no errors in the receiver FIFO or on a read of the ULSR with no remaining receiver FIFO errors. 1b - Set to one when one of the characters in the receiver FIFO encounters an error (framing, parity, or break interrupt)
1 TEM T	Transmitter empty. 0b - Either or both the UTHR or the internal transmitter shift register has a data character. In FIFO mode, a data character is in the transmitter FIFO or the internal transmitter shift register. 1b - Both the UTHR and the internal transmitter shift register are empty. In FIFO mode, both the transmitter FIFO and the internal transmitter shift register are empty.
2 THRE	Transmitter holding register empty. 0b - The UTHR is not empty. 1b - A data character has transferred from the UTHR into the internal transmitter shift register. In FIFO mode, the transmitter FIFO contains no data character.
3 BI	Break interrupt. <b>NOTE:</b> For a single break signal, BI and DR are set multiple times, approximately once every character period. The BI and DR bits continue to be set each character period after they are cleared. This continues for the entire duration of the break signal. To accommodate this behavior, read URBR, which returns zeros and clears DR. Then delay one character period and read URBR again. Note that at the end of the break signal, a random character may be falsely detected and received in the URBR, with ULSR[DR] being set. 0b - This bit is cleared when the ULSR is read or when a valid data transfer is detected (that is, STOP bit is received). 1b - Received data of logic 0 for more than START bit + Data bits + Parity bit + one STOP bits length of time. A new character is not loaded until SIN returns to the mark state (logic 1) and a valid START is detected. In FIFO mode, a zero character is encountered in the FIFO (the zero character is at the top of the FIFO). In FIFO mode, only one zero character is stored.
4 FE	Framing error.

Table continues on the next page...

## DUART register descriptions

Field	Function
	0b - This bit is cleared when ULSR is read or when a new character is loaded into the URBR from the receiver shift register. 1b - Invalid STOP bit for receive data (only the first STOP bit is checked). In FIFO mode, this bit is set when the character that detected a framing error is encountered in the FIFO (that is the character at the top of the FIFO). An attempt to resynchronize occurs after a framing error. The UART assumes that the framing error (due to a logic 0 being read when a logic 1 (STOP) was expected) was due to a STOP bit overlapping with the next START bit, so it assumes this logic 0 sample is a true START bit and then receives the following new data.
5 PE	Parity error. 0b - This bit is cleared when ULSR is read or when a new character is loaded into the URBR. 1b - Unexpected parity value encountered when receiving data. In FIFO mode, the character with the error is at the top of the FIFO .
6 OE	Overrun error. 0b - This bit is cleared when ULSR is read. 1b - Before the URBR is read, the URBR was overwritten with a new character. The old character is lost. In FIFO mode, the receiver FIFO is full (regardless of the receiver FIFO trigger level setting) and a new character has been received into the internal receiver shift register. The old character was overwritten by the new character. Data in the receiver FIFO was not overwritten.
7 DR	Data ready. 0b - This bit is cleared when URBR is read or when all of the data in the receiver FIFO is read. 1b - A character has been received in the URBR or the receiver FIFO.

## 20.4.13 UART modem status register (UMSR1 - UMSR2)

### 20.4.13.1 Offset

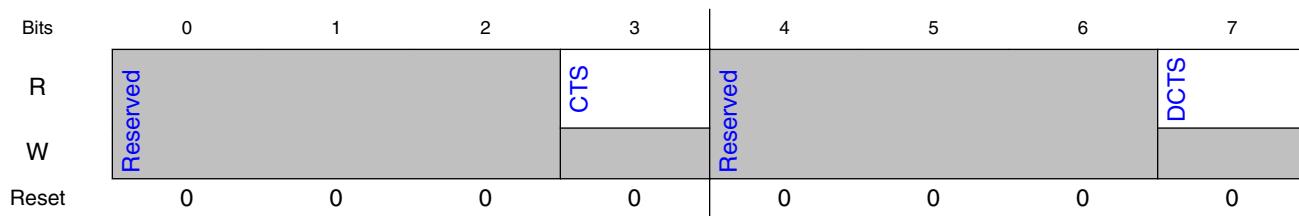
Register	Offset
UMSR1	506h
UMSR2	606h

### 20.4.13.2 Function

This register is accessible when ULCR[DLAB] = x.

The UMSRs track the status of the modem (or external peripheral device) clear to send (CTS\_B) signal for the corresponding UART .

### 20.4.13.3 Diagram



### 20.4.13.4 Fields

Field	Function
0-2 —	Reserved.
3 CTS	Clear to send. Represents the inverted value of the CTS_B input pin from the external peripheral device 0b - Corresponding $UARTn\_CTS\_B$ is negated 1b - Corresponding $UARTn\_CTS\_B$ is asserted. The modem or peripheral device is ready for data transfers.
4-6 —	Reserved.
7 DCTS	Clear to send. 0b - No change on the corresponding $UARTn\_CTS\_B$ signal since the last read of UMSR[CTS] 1b - The $UARTn\_CTS\_B$ value has changed, since the last read of UMSR[CTS]. Causes an interrupt if UIER[EMSI] is set to detect this condition

## 20.4.14 UART scratch register (USCR1 - USCR2)

### 20.4.14.1 Offset

Register	Offset
USCR1	507h
USCR2	607h

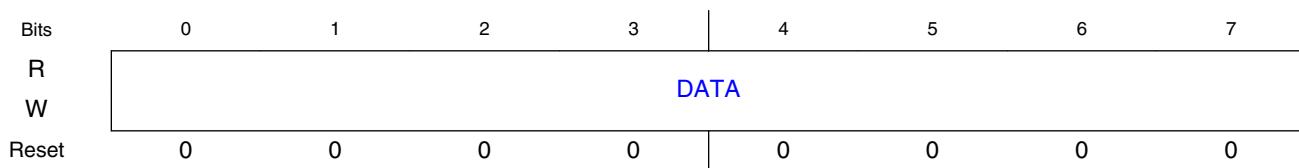
### 20.4.14.2 Function

This register is accessible when ULCR[DLAB] = x.

## DUART register descriptions

The USCR registers are for debugging software or the DUART hardware. The USCRs do not affect the operation of the DUART.

### 20.4.14.3 Diagram



### 20.4.14.4 Fields

Field	Function
0-7	Data
DATA	

## 20.4.15 UART DMA status register (UDSR1 - UDSR2)

### 20.4.15.1 Offset

Register	Offset
UDSR1	510h
UDSR2	610h

### 20.4.15.2 Function

This register is accessible when ULCR[DLAB] = x.

The DMA status registers (UDSRs) are read-only registers that return transmitter and receiver FIFO status. UDSRs also provide the ability to assist DMA data operations to and from the FIFOs.

**Table 20-6. UDSR[TXRDY] Set Conditions**

DMS	FEN	DMA Mode	Meaning
0	0	0	
0	1	0	
1	0	0	
1	1	1	TXRDY is set when the transmitter FIFO is full.

**Table 20-7. UDSR[TXRDY] Cleared Conditions**

DMS	FEN	DMA Mode	Meaning
0	0	0	
0	1	0	
1	0	0	
1	1	1	TXRDY is cleared when there are no characters in the transmitter FIFO or UTHR. TXRDY remains clear when the transmitter FIFO is not yet full.

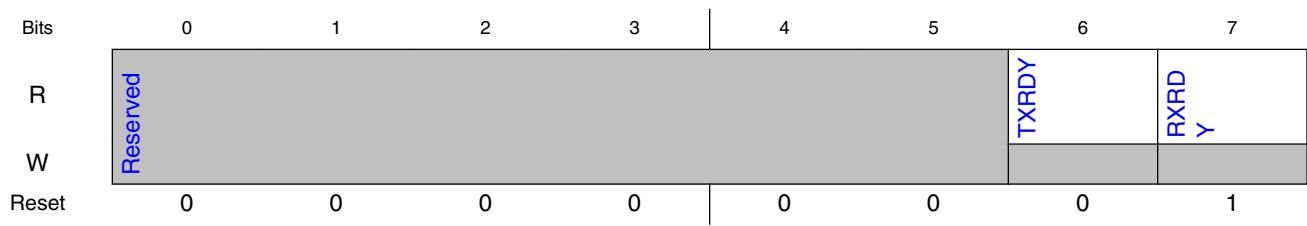
**Table 20-8. UDSR[RXRDY] Set Conditions**

DMS	FEN	DMA Mode	Meaning
0	0	0	
0	1	0	
1	0	0	
1	1	1	RXRDY is set when the trigger level has not been reached and there has been no time out.

**Table 20-9. UDSR[RXRDY] Cleared Conditions**

DMS	FEN	DMA Mode	Meaning
0	0	0	
0	1	0	
1	0	0	
1	1	1	RXRDY is cleared when the trigger level or a time-out has been reached. RXRDY remains cleared until the receiver FIFO is empty.

### 20.4.15.3 Diagram



### 20.4.15.4 Fields

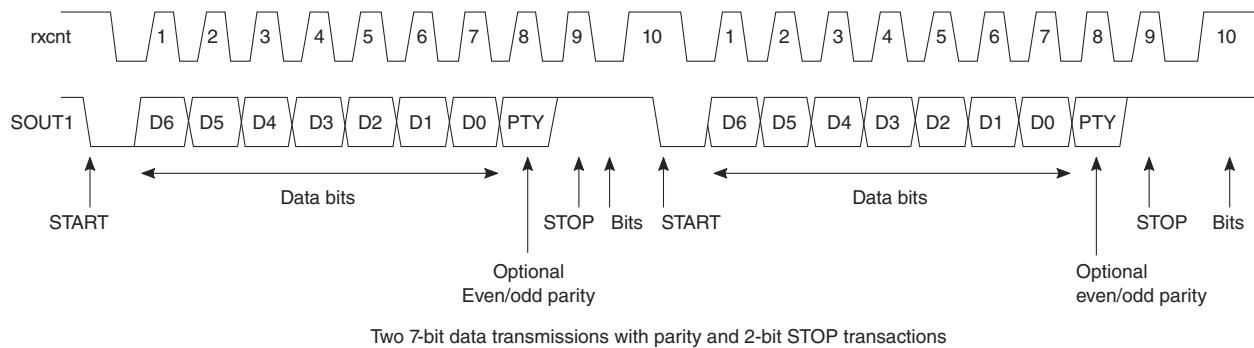
Field	Function
0-5 —	Reserved
6 TXRDY	Transmitter ready. This read-only bit reflects the status of the transmitter FIFO or the UTHR. The status depends on the DMA mode selected, which is determined by the DMS and FEN bits in the UFCR. 0b - This bit is cleared, as shown in <a href="#">Table 20-7</a> . 1b - This bit is set, as shown in <a href="#">Table 20-6</a> .
7 RXRDY	Receiver ready. This read-only bit reflects the status of the receiver FIFO or URBR. The status depends on the DMA mode selected, which is determined by the DMS and FEN bits in the UFCR. 0b - This bit is cleared, as shown in <a href="#">Table 20-9</a> . 1b - This bit is set, as shown in <a href="#">Table 20-8</a> .

## 20.5 Functional description

The following sections provide the function of the DUART controller.

### 20.5.1 Serial interface

The UART bus is a serial, full-duplex, point-to-point bus as shown in the following figure. Therefore, only two devices are attached to the same signals and there is no need for address or arbitration bus cycles.



**Figure 20-2. UART bus interface transaction protocol example**

A standard UART bus transfer is composed of either three or four parts:

- START bit
- Data transfer bits (least-significant bit is first data bit on the bus)
- Parity bit (optional)
- STOP bits

An internal logic sample signal, *rcnt*, uses the frequency of the baud-rate generator to drive the bits on SOUT.

The following sections describe the four components of the serial interface, the baud-rate generator, local loopback mode, different errors, and FIFO mode.

### 20.5.1.1 START bit

A write to the transmitter holding register (UTHR) generates a START bit on the SOUT signal.

Figure 20-2 the figure shows that the START bit is defined as a logic 0. The START bit denotes the beginning of a new data transfer which is limited to the bit length programmed in the UART line control register (ULCR). When the bus is idle, SOUT is high.

### 20.5.1.2 Data transfer

Each data transfer contains 5-8 bits of data.

The ULCR data bit length for the transmitter and receiver UART devices must agree before a transfer begins; otherwise, a parity or framing error may occur. A transfer begins when UTHR is written. At that time a START bit is generated followed by 5-8 of the data

bits previously written to the UTHR. The data bits are driven from the least significant to the most significant bits. After the parity and STOP bits, a new data transfer can begin if new data is written to the UTHR.

### **20.5.1.3 Parity bit**

The user has the option of using even, odd, no parity, or stick parity.

See [UART line control register \(ULCR1 - ULCR2\)](#). Both the receiver and transmitter parity definition must agree before attempting to transfer data. When receiving data, a parity error can occur if an unexpected parity value is detected. See [UART line status register \(ULSR1 - ULSR2\)](#).

### **20.5.1.4 STOP bit**

The transmitter device ends the write transfer by generating a STOP bit.

The STOP bit is always high. The length of the STOP bit(s) can be programmed in the ULCR. Both the receiver and transmitter STOP bit length must agree before attempting to transfer data. A framing error can occur if an invalid STOP bit is detected.

## **20.5.2 Baud-rate generator logic**

Each UART contains an independent programmable baud-rate generator, that is capable of taking the platform clock frequency as input and dividing the input by any divisor from 1 to  $2^{16} - 1$ .

The baud rate is defined as the number of bits per second that can be sent over the UART bus. The formula for calculating baud rate is as follows:

$$\text{Baud rate} = (1/16) \times (\text{platform clock frequency} \div \text{divisor value})$$

Therefore, the output frequency of the baud-rate generator is 16 times the baud rate.

The divisor value is determined by the following two 8-bit registers to form a 16-bit binary number:

- UART divisor most significant byte register (UDMB)
- UART divisor least significant byte register (UDLB)

Upon loading either of the divisor latches, a 16-bit baud-rate counter is loaded.

The divisor latches must be loaded during initialization to ensure proper operation of the baud-rate generator. Both UART devices on the same bus must be programmed for the same baud-rate before starting a transfer.

The baud clock can be passed to the performance monitor by enabling the UAFR[BO] bit. This can be used to determine baud rate errors.

### 20.5.3 Local loopback mode

Local loopback mode is provided for diagnostic testing.

The data written to UTHR can be read from the receiver buffer register (URBR) of the same UART. In this mode, the modem control register UMCR[RTS] is internally tied to the modem status register UMSR[CTS]. The transmitter SOUT is set to a logic 1 and the receiver SIN is disconnected. The output of the transmitter shift register is looped back into the receiver shift register input. The CTS\_B (input signal) is disconnected, RTS\_B is internally connected to CTS\_B, and the RTS\_B (output signal) becomes inactive. In this diagnostic mode, data that is transmitted is immediately received. In local loopback mode, the transmit and receive data paths of the DUART can be verified.

#### NOTE

In local loopback mode, the transmit/receive interrupts are fully operational and can be controlled by the interrupt enable register (UIER).

### 20.5.4 Errors

The following sections describe framing, parity, and overrun errors which may occur while data is transferred on the UART bus.

Each of the error bits are usually cleared, as described below, when the line status register (ULSR) is read.

#### 20.5.4.1 Framing error

When an invalid STOP bit is detected, a framing error occurs and ULSR[FE] is set.

Note that only the first STOP bit is checked. In FIFO mode, ULSR[FE] is set when the character at the top of the FIFO detects a framing error. An attempt to re-synchronize occurs after a framing error. The UART assumes that the framing error (due to a logic 0

being read when a logic 1 (STOP) was expected) was due to a STOP bit overlapping with the next START bit. ULSR[FE] is cleared when ULSR is read or when a new character is loaded into the URBR from the receiver shift register.

#### **20.5.4.2 Parity error**

A parity error occurs, and ULSR[PE] is set, when unexpected parity values are encountered while receiving data.

In FIFO mode, ULSR[PE] is set when the character with the error is at the top of the FIFO. ULSR[PE] is cleared when ULSR is read or when a new character is loaded into the URBR.

#### **20.5.4.3 Overrun error**

When a new (overwriting character) STOP bit is detected and the old character is lost, an overrun error occurs and ULSR[OE] is set.

In FIFO mode, ULSR[OE] is set after the receiver FIFO is full (despite the receiver FIFO trigger level setting) and a new character has been received into the internal receiver shift register. Data in the FIFO is not overwritten; only the shift register data is overwritten. Therefore, the interrupt occurs immediately. ULSR[OE] is cleared when ULSR is read.

### **20.5.5 FIFO mode**

The UARTs use an alternate mode (FIFO mode) to relieve the processor core from excessive software overhead.

The FIFO control register (UFCR) is used to enable and clear the receiver and transmitter FIFOs and set the FIFO receiver trigger level UFCR[RTL] to control the received data available interrupt UIER[ERDAI].

The UFCR also selects the type of DMA signaling. The UDSR[RXRDY] indicates the status of the receiver FIFO. The DMA status registers (UDSR[TXRDY]) indicate when the transmitter FIFO is full. When in FIFO mode, data written to UTHR is placed into the transmitter FIFO. The first byte written to UTHR is the first byte onto the UART bus.

### 20.5.5.1 FIFO interrupts

In FIFO mode, the UIER[ERDAI] is set when a time-out interrupt occurs. When a receive data time-out occurs there is a maskable interrupt condition (through UIER[ERDAI]). See [UART interrupt enable register \(UIER1 - UIER2\)](#) for more details on interrupt enables.

The interrupt ID register (UIIR) indicates if the FIFOs are enabled. The interrupt ID3 bit UIIR[IID3] is only set for FIFO mode interrupts. The character time-out interrupt occurs when no characters have been removed from or input to the receiver FIFO during the last four character times and there is at least one character in the receiver FIFO during this time. The character time-out interrupt (controlled by UIIR[IIDn]) is cleared when the URBR is read. See [UART interrupt ID register \(UIIR1 - UIIR2\)](#) for more information.

The UIIR[FE] bit indicates if FIFO mode is enabled.

### 20.5.5.2 Interrupt control logic

An interrupt is active when DUART interrupt ID register bit 7 (UIIR[IID0]), is cleared.

The interrupt enable register (UIER) is used to mask specific interrupt types. See [UART interrupt enable register \(UIER1 - UIER2\)](#) for more details.

When the interrupts are disabled in UIER, polling software cannot use UIIR[IID0] to determine whether the UART is ready for service. The software must monitor the appropriate bits in the line status (ULSR) and/or the modem status registers (UMSR). UIIR[IID0] can be used for polling if the interrupts are enabled in UIER.

## 20.6 Initialization/Application information

The following requirements must be met for DUART accesses:

- All DUART registers must be mapped to a cache-inhibited and guarded area. (That is, the WIMG setting in the MMU needs to be 0b01X1.)
- All DUART registers are 1 byte wide. Reads and writes to these registers must be byte-wide operations.

A system reset puts the DUART registers to a default state. Before the interface can transfer serial data, the following initialization steps are recommended:

1. Update the programmable interrupt controller (PIC) DUART channel interrupt vector source registers.

## **Initialization/Application information**

2. Set data attributes and control bits in the ULCR, UFCR, UAFR, UMCR, UDLB, and UDMB.
3. Set the data attributes and control bits of the external modem or peripheral device.
4. Set the interrupt enable register (UIER).
5. To start a write transfer, write to the UTHR.
6. Poll UIIR, if the interrupts generated by the DUART are masked.

# **Chapter 21**

## **Enhanced Direct Memory Access (eDMA)**

### **21.1 Overview**

The Enhanced direct memory access (eDMA) is a general purpose DMA which can be used for data transfer between slow peripherals (SPI, I2C, QSPI, FlexTimer, LPUART) and DDR memory.

### **21.2 Introduction**

The enhanced direct memory access (eDMA) controller is a second-generation module capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 32 channels

#### **21.2.1 eDMA system block diagram**

[Figure 21-1](#) illustrates the components of the eDMA system, including the eDMA module ("engine").

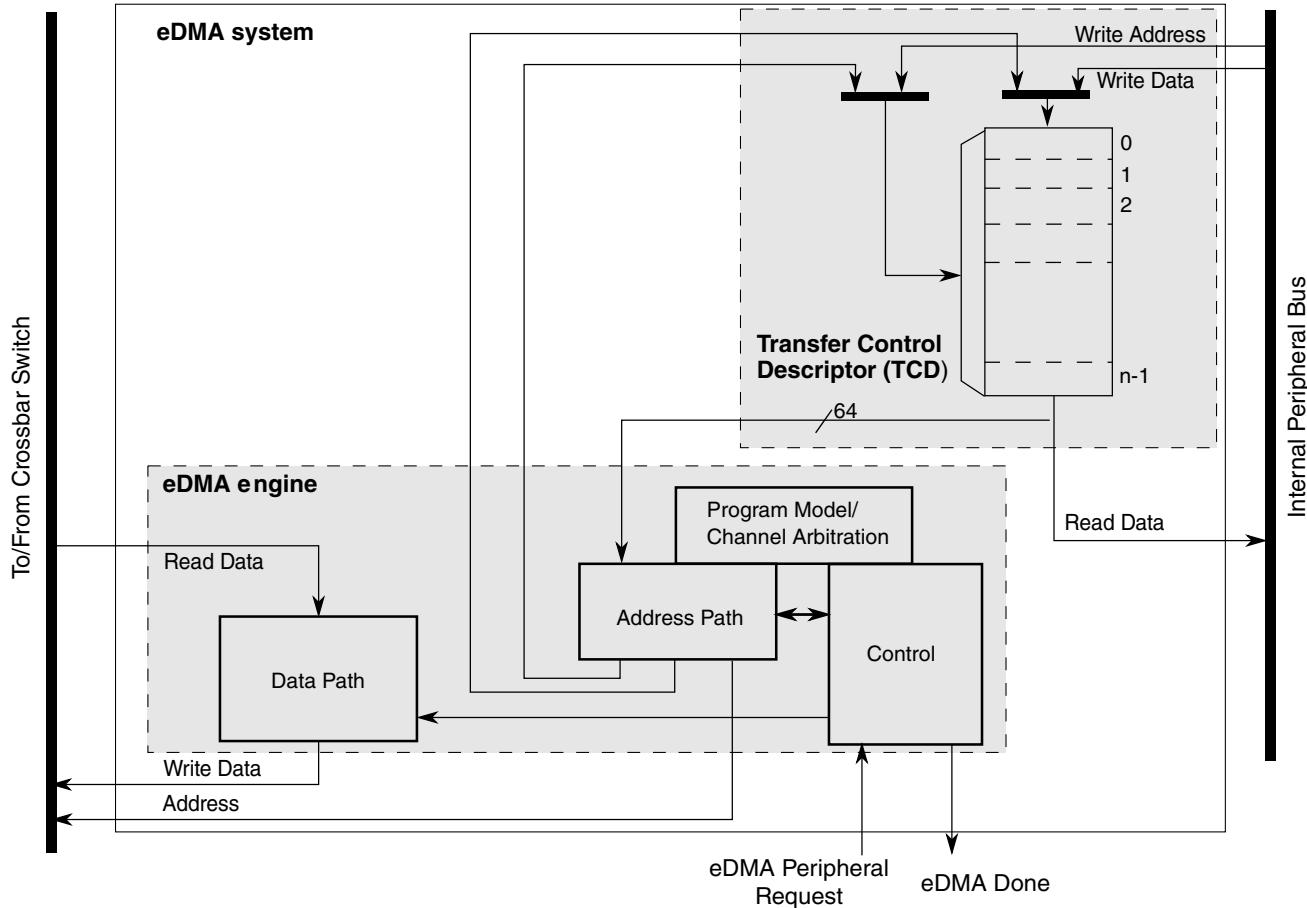


Figure 21-1. eDMA system block diagram

## 21.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 21-1. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, channel x and channel y, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI[n][ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution.  When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes

Table continues on the next page...

**Table 21-1. eDMA engine submodules (continued)**

Submodule	Function
	the new values for the TCDn_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCDn_CITER field, and a possible fetch of the next TCDn from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

**Table 21-2. Transfer control descriptor memory**

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

### 21.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for enhanced addressing modes

## Modes of operation

- 32-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage to support 16- and 32-byte transfers
  - Connections to the crossbar switch for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module,  $n$  is used to reference the channel number.

## 21.3 Modes of operation

The eDMA operates in the following modes:

**Table 21-3. Modes of operation**

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA.  A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	DMA operation is configurable in Debug mode via the control register: <ul style="list-style-type: none"> <li>• If CR[EDBG] is cleared, the DMA continues to operate.</li> <li>• If CR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.</li> </ul>

## 21.4 Memory map/register definition

The eDMA's programming model is partitioned into two regions:

- The first region defines a number of registers providing control functions
- The second region corresponds to the local transfer control descriptor (TCD) memory

### 21.4.1 TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 31. Each  $TCD_n$  definition is presented as 11 registers of 16 or 32 bits.

### 21.4.2 TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

## 21.4.3 TCD structure

### DMA Basics: TCD Structure

- One DMA engine has a number of channels to react to DMA requests
- Each channel has its own TCD

Word Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x1000	SADDR																															
0x1004	SMOD SSIZE DMOD DSIZE SOFF																															
0x1008	NBYTES <sup>1</sup>																															
0x1008	SMLOE <sup>1</sup> DMLOE <sup>1</sup> MLOFF or NBYTES <sup>1</sup> NBYTES <sup>1</sup>																															
0x100C	SLAST																															
0x1010	DADDR																															
0x1014	CITER.E_LINK CITER or CITER.LINKCH CITER DOFF																															
0x1018	DLAST_SGA																															
0x101C	BITER.E_LINK BITER or BITER.LINKCH BITER BWC MAJOR LINKCH DONE ACTIVE MAJOR.E_LINK E_SG D_REQ INT_HALF INT_MAJ START																															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

<sup>1</sup> The fields implemented in Word 2 depend on whether EDMA\_CR[EMLM] bit is set to 0 or 1.

## 21.4.4 Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

## 21.4.5 Endianness

This module's memory map uses big-endian ordering. This means:

- For 8-bit registers, the lower address byte is read as the most significant byte.
- For 16-bit registers, the lower address word is read as the most significant word.

The following figure provides examples of this.

Example 1: 8-bit register structure

Address	Register Data
00h	AAh
01h	BBh
02h	CCh
03h	DDh

For this structure, an 8-bit read of address 00h will yield DDh.

Example 2: 16-bit register structure

Address	Register Data
00h	AABBh
02h	CCDDh

For this structure, a 16-bit read of address 00h will yield CCDDh.

**Figure 21-2. Examples of big-endian register access results**

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_0000	Control Register (DMA_CR)	32	R/W	0000_0400h	<a href="#">21.4.6/837</a>
2C0_0004	Error Status Register (DMA_ES)	32	R	0000_0000h	<a href="#">21.4.7/840</a>
2C0_000C	Enable Request Register (DMA_ERQ)	32	R/W	0000_0000h	<a href="#">21.4.8/842</a>
2C0_0014	Enable Error Interrupt Register (DMA_EEI)	32	R/W	0000_0000h	<a href="#">21.4.9/846</a>
2C0_0018	Clear Enable Error Interrupt Register (DMA_CEEI)	8	W (always reads 0)	00h	<a href="#">21.4.10/849</a>
2C0_0019	Set Enable Error Interrupt Register (DMA_SEEI)	8	W (always reads 0)	00h	<a href="#">21.4.11/850</a>
2C0_001A	Clear Enable Request Register (DMA_CERQ)	8	W (always reads 0)	00h	<a href="#">21.4.12/851</a>
2C0_001B	Set Enable Request Register (DMA_SERQ)	8	W (always reads 0)	00h	<a href="#">21.4.13/852</a>
2C0_001C	Clear DONE Status Bit Register (DMA_CDNE)	8	W (always reads 0)	00h	<a href="#">21.4.14/853</a>
2C0_001D	Set START Bit Register (DMA_SSRT)	8	W (always reads 0)	00h	<a href="#">21.4.15/854</a>
2C0_001E	Clear Error Register (DMA_CERR)	8	W (always reads 0)	00h	<a href="#">21.4.16/855</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_001F	Clear Interrupt Request Register (DMA_CINT)	8	W (always reads 0)	00h	<a href="#">21.4.17/ 856</a>
2C0_0024	Interrupt Request Register (DMA_INT)	32	R/W	0000_0000h	<a href="#">21.4.18/ 857</a>
2C0_002C	Error Register (DMA_ERR)	32	R/W	0000_0000h	<a href="#">21.4.19/ 860</a>
2C0_0034	Hardware Request Status Register (DMA_HRS)	32	R	0000_0000h	<a href="#">21.4.20/ 864</a>
2C0_0100	Channel n Priority Register (DMA_DCHPRI3)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0101	Channel n Priority Register (DMA_DCHPRI2)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0102	Channel n Priority Register (DMA_DCHPRI1)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0103	Channel n Priority Register (DMA_DCHPRI0)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0104	Channel n Priority Register (DMA_DCHPRI7)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0105	Channel n Priority Register (DMA_DCHPRI6)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0106	Channel n Priority Register (DMA_DCHPRI5)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0107	Channel n Priority Register (DMA_DCHPRI4)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0108	Channel n Priority Register (DMA_DCHPRI11)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0109	Channel n Priority Register (DMA_DCHPRI10)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010A	Channel n Priority Register (DMA_DCHPRI9)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010B	Channel n Priority Register (DMA_DCHPRI8)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010C	Channel n Priority Register (DMA_DCHPRI15)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010D	Channel n Priority Register (DMA_DCHPRI14)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010E	Channel n Priority Register (DMA_DCHPRI13)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_010F	Channel n Priority Register (DMA_DCHPRI12)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0110	Channel n Priority Register (DMA_DCHPRI19)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0111	Channel n Priority Register (DMA_DCHPRI18)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>

*Table continues on the next page...*

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_0112	Channel n Priority Register (DMA_DCHPRI17)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0113	Channel n Priority Register (DMA_DCHPRI16)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0114	Channel n Priority Register (DMA_DCHPRI23)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0115	Channel n Priority Register (DMA_DCHPRI22)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0116	Channel n Priority Register (DMA_DCHPRI21)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0117	Channel n Priority Register (DMA_DCHPRI20)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0118	Channel n Priority Register (DMA_DCHPRI27)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_0119	Channel n Priority Register (DMA_DCHPRI26)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011A	Channel n Priority Register (DMA_DCHPRI25)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011B	Channel n Priority Register (DMA_DCHPRI24)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011C	Channel n Priority Register (DMA_DCHPRI31)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011D	Channel n Priority Register (DMA_DCHPRI30)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011E	Channel n Priority Register (DMA_DCHPRI29)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_011F	Channel n Priority Register (DMA_DCHPRI28)	8	R/W	<a href="#">See section</a>	<a href="#">21.4.21/ 870</a>
2C0_1000	TCD Source Address (DMA_TCD0_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1004	TCD Signed Source Address Offset (DMA_TCD0_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1006	TCD Transfer Attributes (DMA_TCD0_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1008	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD0_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1008	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD0_NBYTES_MLOFFNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1008	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD0_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_100C	TCD Last Source Address Adjustment (DMA_TCD0_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1010	TCD Destination Address (DMA_TCD0_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>

*Table continues on the next page...*

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1014	TCD Signed Destination Address Offset (DMA_TCD0_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1016	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1016	DMA_TCD0_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1018	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD0_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_101C	TCD Control and Status (DMA_TCD0_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_101E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_101E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD0_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1020	TCD Source Address (DMA_TCD1_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1024	TCD Signed Source Address Offset (DMA_TCD1_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1026	TCD Transfer Attributes (DMA_TCD1_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1028	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD1_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1028	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD1_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1028	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD1_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_102C	TCD Last Source Address Adjustment (DMA_TCD1_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1030	TCD Destination Address (DMA_TCD1_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1034	TCD Signed Destination Address Offset (DMA_TCD1_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1036	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1036	DMA_TCD1_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1038	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD1_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_103C	TCD Control and Status (DMA_TCD1_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_103E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_103E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD1_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1040	TCD Source Address (DMA_TCD2_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1044	TCD Signed Source Address Offset (DMA_TCD2_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1046	TCD Transfer Attributes (DMA_TCD2_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1048	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD2_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1048	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD2_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1048	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD2_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_104C	TCD Last Source Address Adjustment (DMA_TCD2_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1050	TCD Destination Address (DMA_TCD2_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1054	TCD Signed Destination Address Offset (DMA_TCD2_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1056	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1056	DMA_TCD2_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1058	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD2_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_105C	TCD Control and Status (DMA_TCD2_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_105E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_105E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD2_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1060	TCD Source Address (DMA_TCD3_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1064	TCD Signed Source Address Offset (DMA_TCD3_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1066	TCD Transfer Attributes (DMA_TCD3_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1068	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD3_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1068	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD3_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>

*Table continues on the next page...*

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1068	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD3_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_106C	TCD Last Source Address Adjustment (DMA_TCD3_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1070	TCD Destination Address (DMA_TCD3_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1074	TCD Signed Destination Address Offset (DMA_TCD3_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1076	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1076	DMA_TCD3_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1078	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD3_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_107C	TCD Control and Status (DMA_TCD3_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_107E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_107E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD3_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1080	TCD Source Address (DMA_TCD4_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1084	TCD Signed Source Address Offset (DMA_TCD4_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1086	TCD Transfer Attributes (DMA_TCD4_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1088	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD4_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1088	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD4_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1088	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD4_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_108C	TCD Last Source Address Adjustment (DMA_TCD4_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1090	TCD Destination Address (DMA_TCD4_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1094	TCD Signed Destination Address Offset (DMA_TCD4_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1096	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1096	DMA_TCD4_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_1098	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD4_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_109C	TCD Control and Status (DMA_TCD4_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_109E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_109E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD4_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_10A0	TCD Source Address (DMA_TCD5_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_10A4	TCD Signed Source Address Offset (DMA_TCD5_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_10A6	TCD Transfer Attributes (DMA_TCD5_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_10A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD5_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_10A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD5_NBYTES_MLOFFNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_10A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD5_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_10AC	TCD Last Source Address Adjustment (DMA_TCD5_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_10B0	TCD Destination Address (DMA_TCD5_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_10B4	TCD Signed Destination Address Offset (DMA_TCD5_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_10B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_10B6	DMA_TCD5_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_10B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD5_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_10BC	TCD Control and Status (DMA_TCD5_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_10BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_10BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD5_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_10C0	TCD Source Address (DMA_TCD6_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_10C4	TCD Signed Source Address Offset (DMA_TCD6_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_10C6	TCD Transfer Attributes (DMA_TCD6_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_10C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD6_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_10C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD6_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_10C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD6_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_10CC	TCD Last Source Address Adjustment (DMA_TCD6_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_10D0	TCD Destination Address (DMA_TCD6_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_10D4	TCD Signed Destination Address Offset (DMA_TCD6_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_10D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_10D6	DMA_TCD6_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_10D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD6_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_10DC	TCD Control and Status (DMA_TCD6_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_10DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_10DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD6_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_10E0	TCD Source Address (DMA_TCD7_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_10E4	TCD Signed Source Address Offset (DMA_TCD7_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_10E6	TCD Transfer Attributes (DMA_TCD7_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_10E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD7_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_10E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD7_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_10E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD7_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_10EC	TCD Last Source Address Adjustment (DMA_TCD7_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_10F0	TCD Destination Address (DMA_TCD7_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_10F4	TCD Signed Destination Address Offset (DMA_TCD7_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_10F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_10F6	DMA_TCD7_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_10F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD7_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_10FC	TCD Control and Status (DMA_TCD7_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_10FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_10FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD7_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1100	TCD Source Address (DMA_TCD8_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1104	TCD Signed Source Address Offset (DMA_TCD8_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1106	TCD Transfer Attributes (DMA_TCD8_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1108	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD8_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1108	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD8_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1108	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD8_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_110C	TCD Last Source Address Adjustment (DMA_TCD8_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1110	TCD Destination Address (DMA_TCD8_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1114	TCD Signed Destination Address Offset (DMA_TCD8_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1116	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1116	DMA_TCD8_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1118	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD8_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_111C	TCD Control and Status (DMA_TCD8_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_111E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_111E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD8_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1120	TCD Source Address (DMA_TCD9_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1124	TCD Signed Source Address Offset (DMA_TCD9_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1126	TCD Transfer Attributes (DMA_TCD9_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1128	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD9_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1128	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD9_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1128	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD9_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_112C	TCD Last Source Address Adjustment (DMA_TCD9_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1130	TCD Destination Address (DMA_TCD9_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1134	TCD Signed Destination Address Offset (DMA_TCD9_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1136	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1136	DMA_TCD9_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1138	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD9_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_113C	TCD Control and Status (DMA_TCD9_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_113E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_113E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD9_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1140	TCD Source Address (DMA_TCD10_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1144	TCD Signed Source Address Offset (DMA_TCD10_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1146	TCD Transfer Attributes (DMA_TCD10_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1148	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD10_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1148	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD10_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_1148	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD10_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_114C	TCD Last Source Address Adjustment (DMA_TCD10_Slast)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1150	TCD Destination Address (DMA_TCD10_Daddr)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1154	TCD Signed Destination Address Offset (DMA_TCD10_Doff)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1156	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_Citer_elinkyes)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1156	DMA_TCD10_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1158	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD10_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_115C	TCD Control and Status (DMA_TCD10_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_115E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_Biter_elinkyes)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_115E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD10_Biter_elinkno)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1160	TCD Source Address (DMA_TCD11_Saddr)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1164	TCD Signed Source Address Offset (DMA_TCD11_Soff)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1166	TCD Transfer Attributes (DMA_TCD11_Attr)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1168	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD11_Nbytes_mlno)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1168	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD11_Nbytes_mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1168	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD11_Nbytes_mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_116C	TCD Last Source Address Adjustment (DMA_TCD11_Slast)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1170	TCD Destination Address (DMA_TCD11_Daddr)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1174	TCD Signed Destination Address Offset (DMA_TCD11_Doff)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1176	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_Citer_elinkyes)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1176	DMA_TCD11_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1178	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD11_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_117C	TCD Control and Status (DMA_TCD11_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_117E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_117E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD11_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1180	TCD Source Address (DMA_TCD12_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1184	TCD Signed Source Address Offset (DMA_TCD12_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1186	TCD Transfer Attributes (DMA_TCD12_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1188	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD12_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1188	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD12_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1188	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD12_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_118C	TCD Last Source Address Adjustment (DMA_TCD12_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1190	TCD Destination Address (DMA_TCD12_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1194	TCD Signed Destination Address Offset (DMA_TCD12_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1196	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1196	DMA_TCD12_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1198	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD12_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_119C	TCD Control and Status (DMA_TCD12_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_119E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_119E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD12_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_11A0	TCD Source Address (DMA_TCD13_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2C0_11A4	TCD Signed Source Address Offset (DMA_TCD13_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_11A6	TCD Transfer Attributes (DMA_TCD13_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_11A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD13_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_11A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD13_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_11A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD13_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_11AC	TCD Last Source Address Adjustment (DMA_TCD13_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_11B0	TCD Destination Address (DMA_TCD13_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_11B4	TCD Signed Destination Address Offset (DMA_TCD13_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_11B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_11B6	DMA_TCD13_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_11B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD13_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_11BC	TCD Control and Status (DMA_TCD13_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_11BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_11BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD13_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_11C0	TCD Source Address (DMA_TCD14_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_11C4	TCD Signed Source Address Offset (DMA_TCD14_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_11C6	TCD Transfer Attributes (DMA_TCD14_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_11C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD14_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_11C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD14_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_11C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD14_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_11CC	TCD Last Source Address Adjustment (DMA_TCD14_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_11D0	TCD Destination Address (DMA_TCD14_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_11D4	TCD Signed Destination Address Offset (DMA_TCD14_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_11D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_11D6	DMA_TCD14_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_11D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD14_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_11DC	TCD Control and Status (DMA_TCD14_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_11DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_11DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD14_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_11E0	TCD Source Address (DMA_TCD15_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_11E4	TCD Signed Source Address Offset (DMA_TCD15_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_11E6	TCD Transfer Attributes (DMA_TCD15_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_11E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD15_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_11E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD15_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_11E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD15_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_11EC	TCD Last Source Address Adjustment (DMA_TCD15_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_11F0	TCD Destination Address (DMA_TCD15_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_11F4	TCD Signed Destination Address Offset (DMA_TCD15_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_11F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_11F6	DMA_TCD15_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_11F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD15_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_11FC	TCD Control and Status (DMA_TCD15_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_11FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_11FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD15_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1200	TCD Source Address (DMA_TCD16_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1204	TCD Signed Source Address Offset (DMA_TCD16_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1206	TCD Transfer Attributes (DMA_TCD16_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1208	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD16_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1208	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD16_NBYTES_MLOFFNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1208	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD16_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_120C	TCD Last Source Address Adjustment (DMA_TCD16_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1210	TCD Destination Address (DMA_TCD16_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1214	TCD Signed Destination Address Offset (DMA_TCD16_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1216	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD16_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1216	DMA_TCD16_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1218	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD16_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_121C	TCD Control and Status (DMA_TCD16_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_121E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD16_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_121E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD16_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1220	TCD Source Address (DMA_TCD17_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1224	TCD Signed Source Address Offset (DMA_TCD17_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1226	TCD Transfer Attributes (DMA_TCD17_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1228	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD17_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1228	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD17_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1228	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD17_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_122C	TCD Last Source Address Adjustment (DMA_TCD17_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1230	TCD Destination Address (DMA_TCD17_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1234	TCD Signed Destination Address Offset (DMA_TCD17_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1236	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD17_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1236	DMA_TCD17_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1238	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD17_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_123C	TCD Control and Status (DMA_TCD17_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_123E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD17_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_123E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD17_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1240	TCD Source Address (DMA_TCD18_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1244	TCD Signed Source Address Offset (DMA_TCD18_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1246	TCD Transfer Attributes (DMA_TCD18_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1248	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD18_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1248	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD18_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1248	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD18_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_124C	TCD Last Source Address Adjustment (DMA_TCD18_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1250	TCD Destination Address (DMA_TCD18_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1254	TCD Signed Destination Address Offset (DMA_TCD18_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1256	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD18_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1256	DMA_TCD18_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1258	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD18_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_125C	TCD Control and Status (DMA_TCD18_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_125E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD18_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_125E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD18_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1260	TCD Source Address (DMA_TCD19_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1264	TCD Signed Source Address Offset (DMA_TCD19_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1266	TCD Transfer Attributes (DMA_TCD19_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1268	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD19_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1268	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD19_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1268	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD19_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_126C	TCD Last Source Address Adjustment (DMA_TCD19_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1270	TCD Destination Address (DMA_TCD19_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1274	TCD Signed Destination Address Offset (DMA_TCD19_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1276	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD19_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1276	DMA_TCD19_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1278	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD19_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_127C	TCD Control and Status (DMA_TCD19_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>

*Table continues on the next page...*

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_127E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD19_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_127E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD19_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_1280	TCD Source Address (DMA_TCD20_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_1284	TCD Signed Source Address Offset (DMA_TCD20_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_1286	TCD Transfer Attributes (DMA_TCD20_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>
2C0_1288	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD20_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/873</a>
2C0_1288	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD20_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/874</a>
2C0_1288	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD20_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/875</a>
2C0_128C	TCD Last Source Address Adjustment (DMA_TCD20_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/877</a>
2C0_1290	TCD Destination Address (DMA_TCD20_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/877</a>
2C0_1294	TCD Signed Destination Address Offset (DMA_TCD20_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/878</a>
2C0_1296	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD20_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/878</a>
2C0_1296	DMA_TCD20_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/879</a>
2C0_1298	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD20_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/880</a>
2C0_129C	TCD Control and Status (DMA_TCD20_CSR)	16	R/W	Undefined	<a href="#">21.4.34/881</a>
2C0_129E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD20_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_129E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD20_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_12A0	TCD Source Address (DMA_TCD21_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_12A4	TCD Signed Source Address Offset (DMA_TCD21_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_12A6	TCD Transfer Attributes (DMA_TCD21_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_12A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD21_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_12A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD21_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_12A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD21_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_12AC	TCD Last Source Address Adjustment (DMA_TCD21_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_12B0	TCD Destination Address (DMA_TCD21_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_12B4	TCD Signed Destination Address Offset (DMA_TCD21_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_12B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD21_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_12B6	DMA_TCD21_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_12B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD21_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_12BC	TCD Control and Status (DMA_TCD21_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_12BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD21_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_12BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD21_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_12C0	TCD Source Address (DMA_TCD22_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_12C4	TCD Signed Source Address Offset (DMA_TCD22_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_12C6	TCD Transfer Attributes (DMA_TCD22_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_12C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD22_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_12C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD22_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_12C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD22_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_12CC	TCD Last Source Address Adjustment (DMA_TCD22_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_12D0	TCD Destination Address (DMA_TCD22_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_12D4	TCD Signed Destination Address Offset (DMA_TCD22_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_12D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD22_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_12D6	DMA_TCD22_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_12D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD22_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_12DC	TCD Control and Status (DMA_TCD22_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_12DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD22_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_12DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD22_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_12E0	TCD Source Address (DMA_TCD23_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_12E4	TCD Signed Source Address Offset (DMA_TCD23_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_12E6	TCD Transfer Attributes (DMA_TCD23_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_12E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD23_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_12E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD23_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_12E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD23_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_12EC	TCD Last Source Address Adjustment (DMA_TCD23_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_12F0	TCD Destination Address (DMA_TCD23_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_12F4	TCD Signed Destination Address Offset (DMA_TCD23_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_12F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD23_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_12F6	DMA_TCD23_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_12F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD23_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_12FC	TCD Control and Status (DMA_TCD23_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_12FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD23_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_12FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD23_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_1300	TCD Source Address (DMA_TCD24_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_1304	TCD Signed Source Address Offset (DMA_TCD24_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_1306	TCD Transfer Attributes (DMA_TCD24_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>
2C0_1308	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD24_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/873</a>
2C0_1308	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD24_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/874</a>
2C0_1308	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD24_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/875</a>
2C0_130C	TCD Last Source Address Adjustment (DMA_TCD24_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/877</a>
2C0_1310	TCD Destination Address (DMA_TCD24_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/877</a>
2C0_1314	TCD Signed Destination Address Offset (DMA_TCD24_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/878</a>
2C0_1316	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD24_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/878</a>
2C0_1316	DMA_TCD24_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/879</a>
2C0_1318	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD24_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/880</a>
2C0_131C	TCD Control and Status (DMA_TCD24_CSR)	16	R/W	Undefined	<a href="#">21.4.34/881</a>
2C0_131E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD24_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_131E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD24_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_1320	TCD Source Address (DMA_TCD25_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_1324	TCD Signed Source Address Offset (DMA_TCD25_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_1326	TCD Transfer Attributes (DMA_TCD25_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1328	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD25_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1328	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD25_NBYTES_MLOFFNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1328	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD25_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_132C	TCD Last Source Address Adjustment (DMA_TCD25_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1330	TCD Destination Address (DMA_TCD25_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1334	TCD Signed Destination Address Offset (DMA_TCD25_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1336	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD25_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1336	DMA_TCD25_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1338	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD25_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_133C	TCD Control and Status (DMA_TCD25_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_133E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD25_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_133E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD25_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1340	TCD Source Address (DMA_TCD26_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1344	TCD Signed Source Address Offset (DMA_TCD26_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1346	TCD Transfer Attributes (DMA_TCD26_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1348	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD26_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1348	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD26_NBYTES_MLOFFNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1348	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD26_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_134C	TCD Last Source Address Adjustment (DMA_TCD26_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1350	TCD Destination Address (DMA_TCD26_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_1354	TCD Signed Destination Address Offset (DMA_TCD26_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1356	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD26_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1356	DMA_TCD26_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1358	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD26_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_135C	TCD Control and Status (DMA_TCD26_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_135E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD26_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_135E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD26_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_1360	TCD Source Address (DMA_TCD27_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_1364	TCD Signed Source Address Offset (DMA_TCD27_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_1366	TCD Transfer Attributes (DMA_TCD27_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_1368	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD27_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_1368	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD27_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_1368	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD27_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_136C	TCD Last Source Address Adjustment (DMA_TCD27_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_1370	TCD Destination Address (DMA_TCD27_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_1374	TCD Signed Destination Address Offset (DMA_TCD27_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_1376	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD27_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_1376	DMA_TCD27_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_1378	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD27_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_137C	TCD Control and Status (DMA_TCD27_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>

*Table continues on the next page...*

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_137E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD27_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_137E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD27_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_1380	TCD Source Address (DMA_TCD28_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_1384	TCD Signed Source Address Offset (DMA_TCD28_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_1386	TCD Transfer Attributes (DMA_TCD28_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>
2C0_1388	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD28_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/873</a>
2C0_1388	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD28_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/874</a>
2C0_1388	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD28_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/875</a>
2C0_138C	TCD Last Source Address Adjustment (DMA_TCD28_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/877</a>
2C0_1390	TCD Destination Address (DMA_TCD28_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/877</a>
2C0_1394	TCD Signed Destination Address Offset (DMA_TCD28_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/878</a>
2C0_1396	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD28_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/878</a>
2C0_1396	DMA_TCD28_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/879</a>
2C0_1398	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD28_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/880</a>
2C0_139C	TCD Control and Status (DMA_TCD28_CSR)	16	R/W	Undefined	<a href="#">21.4.34/881</a>
2C0_139E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD28_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_139E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD28_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>
2C0_13A0	TCD Source Address (DMA_TCD29_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/871</a>
2C0_13A4	TCD Signed Source Address Offset (DMA_TCD29_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/872</a>
2C0_13A6	TCD Transfer Attributes (DMA_TCD29_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/872</a>

Table continues on the next page...

## DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_13A8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD29_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_13A8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD29_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_13A8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD29_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_13AC	TCD Last Source Address Adjustment (DMA_TCD29_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_13B0	TCD Destination Address (DMA_TCD29_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_13B4	TCD Signed Destination Address Offset (DMA_TCD29_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_13B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD29_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_13B6	DMA_TCD29_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_13B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD29_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_13BC	TCD Control and Status (DMA_TCD29_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_13BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD29_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_13BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD29_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_13C0	TCD Source Address (DMA_TCD30_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_13C4	TCD Signed Source Address Offset (DMA_TCD30_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_13C6	TCD Transfer Attributes (DMA_TCD30_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_13C8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD30_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_13C8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD30_NBYTES_Mloffno)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_13C8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD30_NBYTES_Mloffyes)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_13CC	TCD Last Source Address Adjustment (DMA_TCD30_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_13D0	TCD Destination Address (DMA_TCD30_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>

Table continues on the next page...

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_13D4	TCD Signed Destination Address Offset (DMA_TCD30_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_13D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD30_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_13D6	DMA_TCD30_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_13D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD30_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_13DC	TCD Control and Status (DMA_TCD30_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>
2C0_13DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD30_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/ 883</a>
2C0_13DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD30_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/ 884</a>
2C0_13E0	TCD Source Address (DMA_TCD31_SADDR)	32	R/W	Undefined	<a href="#">21.4.22/ 871</a>
2C0_13E4	TCD Signed Source Address Offset (DMA_TCD31_SOFF)	16	R/W	Undefined	<a href="#">21.4.23/ 872</a>
2C0_13E6	TCD Transfer Attributes (DMA_TCD31_ATTR)	16	R/W	Undefined	<a href="#">21.4.24/ 872</a>
2C0_13E8	TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA_TCD31_NBYTES_MLNO)	32	R/W	Undefined	<a href="#">21.4.25/ 873</a>
2C0_13E8	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA_TCD31_NBYTES_MloffNO)	32	R/W	Undefined	<a href="#">21.4.26/ 874</a>
2C0_13E8	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA_TCD31_NBYTES_MloffYES)	32	R/W	Undefined	<a href="#">21.4.27/ 875</a>
2C0_13EC	TCD Last Source Address Adjustment (DMA_TCD31_SLAST)	32	R/W	Undefined	<a href="#">21.4.28/ 877</a>
2C0_13F0	TCD Destination Address (DMA_TCD31_DADDR)	32	R/W	Undefined	<a href="#">21.4.29/ 877</a>
2C0_13F4	TCD Signed Destination Address Offset (DMA_TCD31_DOFF)	16	R/W	Undefined	<a href="#">21.4.30/ 878</a>
2C0_13F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD31_CITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.31/ 878</a>
2C0_13F6	DMA_TCD31_CITER_ELINKNO	16	R/W	Undefined	<a href="#">21.4.32/ 879</a>
2C0_13F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD31_DLASTSGA)	32	R/W	Undefined	<a href="#">21.4.33/ 880</a>
2C0_13FC	TCD Control and Status (DMA_TCD31_CSR)	16	R/W	Undefined	<a href="#">21.4.34/ 881</a>

*Table continues on the next page...*

**DMA memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2C0_13FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD31_BITER_ELINKYES)	16	R/W	Undefined	<a href="#">21.4.35/883</a>
2C0_13FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD31_BITER_ELINKNO)	16	R/W	Undefined	<a href="#">21.4.36/884</a>

**21.4.6 Control Register (DMA\_CR)**

The CR defines the basic operating configuration of the DMA. The DMA arbitrates channel service requests in two groups of 16 channels each:

- Group 1 contains channels 31-16
- Group 0 contains channels 15-0

Arbitration within a group can be configured to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, the highest priority channel requesting service is selected to execute. The channel priority registers assign the priorities; see the DCHPRIn registers. For round-robin arbitration, the channel priorities are ignored and channels within each group are cycled through (from high to low channel number) without regard to priority.

**NOTE**

For correct operation, writes to the CR register must be performed only when the DMA channels are inactive; that is, when TCDn\_CSR[ACTIVE] bits are cleared.

The group priorities operate in a similar fashion. In group fixed priority arbitration mode, channel service requests in the highest priority group are executed first, where priority level 1 is the highest and priority level 0 is the lowest. The group priorities are assigned in the GRPnPRI fields of the DMA Control Register (CR). All group priorities must have unique values prior to any channel service requests occurring; otherwise, a configuration error will be reported. For group round robin arbitration, the group priorities are ignored and the groups are cycled through (from high to low group number) without regard to priority.

Minor loop offsets are address offset values added to the final source address (TCDn\_SADDR) or destination address (TCDn\_DADDR) upon minor loop completion. When minor loop offsets are enabled, the minor loop offset (Mloff) is added to the final source address (TCDn\_SADDR), to the final destination address (TCDn\_DADDR),

## Memory map/register definition

or to both prior to the addresses being written back into the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn\_SLAST and TCDn\_DLAST\_SGA) are used to compute the next TCDn\_SADDR and TCDn\_DADDR values.

When minor loop mapping is enabled (EMLM is 1), TCDn word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn\_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn\_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

When minor loop mapping is disabled (EMLM is 0), all 32 bits of TCDn word2 are assigned to the NBYTES field.

Address: 2C0\_0000h base + 0h offset = 2C0\_0000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W															CX	ECX
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				0		GRPRI	0	GRPOPRI	EMLM	CLM	HALT	HOE	ERGA	ERCA	EDBG	Reserved
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

### DMA\_CR field descriptions

Field	Description
0–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
14 CX	Cancel Transfer 0 Normal operation 1 Cancel the remaining data transfer. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The CX bit clears itself after the cancel has been honored. This cancel retires the channel normally as if the minor loop was completed.

Table continues on the next page...

**DMA\_CR field descriptions (continued)**

Field	Description
15 ECX	Error Cancel Transfer  0 Normal operation 1 Cancel the remaining data transfer in the same fashion as the CX bit. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The ECX bit clears itself after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating the Error Status register (DMAx_ES) and generating an optional error interrupt.
16–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 GRP1PRI	Channel Group 1 Priority  Group 1 priority level when fixed priority group arbitration is enabled.
22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 GRP0PRI	Channel Group 0 Priority  Group 0 priority level when fixed priority group arbitration is enabled.
24 EMLM	Enable Minor Loop Mapping  0 Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1 Enabled. TCDn.word2 is redefined to include individual enable fields, an offset field, and the NBYTES field. The individual enable fields allow the minor loop offset to be applied to the source address, the destination address, or both. The NBYTES field is reduced when either offset is enabled.
25 CLM	Continuous Link Mode  <b>NOTE:</b> Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, e.g., if the channel's NBYTES value is the same as either the source or destination size. The same data transfer profile can be achieved by simply increasing the NBYTES value, which provides more efficient, faster processing.  0 A minor loop channel link made to itself goes through channel arbitration before being activated again. 1 A minor loop channel link made to itself does not go through channel arbitration before being activated again. Upon minor loop completion, the channel activates again if that channel has a minor loop channel link enabled and the link channel is itself. This effectively applies the minor loop offsets and restarts the next minor loop.
26 HALT	Halt DMA Operations  0 Normal operation 1 Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.
27 HOE	Halt On Error  0 Normal operation 1 Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
28 ERGA	Enable Round Robin Group Arbitration  0 Fixed priority arbitration is used for selection among the groups. 1 Round robin arbitration is used for selection among the groups.

*Table continues on the next page...*

**DMA\_CR field descriptions (continued)**

Field	Description
29 ERCA	Enable Round Robin Channel Arbitration 0 Fixed priority arbitration is used for channel selection within each group. 1 Round robin arbitration is used for channel selection within each group.
30 EDBG	Enable Debug 0 When in debug mode, the DMA continues to operate. 1 When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
31 Reserved	This field is reserved. Reserved

**21.4.7 Error Status Register (DMA\_ES)**

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor, or
  - An illegal priority register setting in fixed-arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error bit that will be set when a transfer is canceled via the corresponding cancel transfer control bit

See [Fault reporting and handling](#) for more details.

Address: 2C0\_0000h base + 4h offset = 2C0\_0004h

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R	VLD								0								ECX
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R	GPE	CPE	0		ERRCHN					SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**DMA\_ES field descriptions**

Field	Description
0 VLD	Logical OR of all ERR status bits

*Table continues on the next page...*

**DMA\_ES field descriptions (continued)**

Field	Description
	<p>0 No ERR bits are set.</p> <p>1 At least one ERR bit is set indicating a valid error exists that has not been cleared.</p>
1–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 ECX	<p>Transfer Canceled</p> <p>0 No canceled transfers</p> <p>1 The last recorded entry was a canceled transfer by the error cancel transfer input</p>
16 GPE	<p>Group Priority Error</p> <p>0 No group priority error</p> <p>1 The last recorded error was a configuration error among the group priorities. All group priorities are not unique.</p>
17 CPE	<p>Channel Priority Error</p> <p>0 No channel priority error</p> <p>1 The last recorded error was a configuration error in the channel priorities within a group. Channel priorities within a group are not unique.</p>
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–23 ERRCHN	<p>Error Channel Number or Canceled Channel Number</p> <p>The channel number of the last recorded error, excluding GPE and CPE errors, or last recorded error canceled transfer.</p>
24 SAE	<p>Source Address Error</p> <p>0 No source address configuration error.</p> <p>1 The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].</p>
25 SOE	<p>Source Offset Error</p> <p>0 No source offset configuration error</p> <p>1 The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].</p>
26 DAE	<p>Destination Address Error</p> <p>0 No destination address configuration error</p> <p>1 The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].</p>
27 DOE	<p>Destination Offset Error</p> <p>0 No destination offset configuration error</p> <p>1 The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].</p>
28 NCE	<p>NBYTES/CITER Configuration Error</p> <p>0 No NBYTES/CITER configuration error</p> <p>1 The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields.</p> <ul style="list-style-type: none"> <li>• TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], or</li> </ul>

*Table continues on the next page...*

**DMA\_ES field descriptions (continued)**

Field	Description
	<ul style="list-style-type: none"> <li>TCDn_CITER[CITER] is equal to zero, or</li> <li>TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]</li> </ul>
29 SGE	<p>Scatter/Gather Configuration Error</p> <p>0 No scatter/gather configuration error</p> <p>1 The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.</p>
30 SBE	<p>Source Bus Error</p> <p>0 No source bus error</p> <p>1 The last recorded error was a bus error on a source read</p>
31 DBE	<p>Destination Bus Error</p> <p>0 No destination bus error</p> <p>1 The last recorded error was a bus error on a destination write</p>

**21.4.8 Enable Request Register (DMA\_ERQ)**

The ERQ register provides a bit map for the 32 channels to enable the request signal for each channel. The state of any given channel enable is directly affected by writes to this register; it is also affected by writes to the SERQ and CERQ registers. These registers are provided so the request enable for a single channel can easily be modified without needing to perform a read-modify-write sequence to the ERQ.

DMA request input signals and this enable request flag must be asserted before a channel's hardware service request is accepted. The state of the DMA enable request flag does not affect a channel service request made explicitly through software or a linked channel request.

**NOTE**

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 2C0\_0000h base + Ch offset = 2C0\_000Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ERQ31	ERQ30	ERQ29	ERQ28	ERQ27	ERQ26	ERQ25	ERQ24	ERQ23	ERQ22	ERQ21	ERQ20	ERQ19	ERQ18	ERQ17	ERQ16
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ERQ15	ERQ14	ERQ13	ERQ12	ERQ11	ERQ10	ERQ9	ERQ8	ERQ7	ERQ6	ERQ5	ERQ4	ERQ3	ERQ2	ERQ1	ERQ0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DMA\_ERQ field descriptions**

Field	Description
0 ERQ31	Enable DMA Request 31  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
1 ERQ30	Enable DMA Request 30  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
2 ERQ29	Enable DMA Request 29  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
3 ERQ28	Enable DMA Request 28  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
4 ERQ27	Enable DMA Request 27  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
5 ERQ26	Enable DMA Request 26  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
6 ERQ25	Enable DMA Request 25  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
7 ERQ24	Enable DMA Request 24  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
8 ERQ23	Enable DMA Request 23  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
9 ERQ22	Enable DMA Request 22  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

*Table continues on the next page...*

**DMA\_ERQ field descriptions (continued)**

Field	Description
10 ERQ21	Enable DMA Request 21  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
11 ERQ20	Enable DMA Request 20  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
12 ERQ19	Enable DMA Request 19  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
13 ERQ18	Enable DMA Request 18  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
14 ERQ17	Enable DMA Request 17  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
15 ERQ16	Enable DMA Request 16  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
16 ERQ15	Enable DMA Request 15  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
17 ERQ14	Enable DMA Request 14  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
18 ERQ13	Enable DMA Request 13  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
19 ERQ12	Enable DMA Request 12  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
20 ERQ11	Enable DMA Request 11  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
21 ERQ10	Enable DMA Request 10  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

*Table continues on the next page...*

**DMA\_ERQ field descriptions (continued)**

Field	Description
22 ERQ9	Enable DMA Request 9  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
23 ERQ8	Enable DMA Request 8  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
24 ERQ7	Enable DMA Request 7  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
25 ERQ6	Enable DMA Request 6  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
26 ERQ5	Enable DMA Request 5  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
27 ERQ4	Enable DMA Request 4  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
28 ERQ3	Enable DMA Request 3  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
29 ERQ2	Enable DMA Request 2  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
30 ERQ1	Enable DMA Request 1  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
31 ERQ0	Enable DMA Request 0  0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

## 21.4.9 Enable Error Interrupt Register (DMA\_EEI)

The EEI register provides a bit map for the 32 channels to enable the error interrupt signal for each channel. The state of any given channel's error interrupt enable is directly affected by writes to this register; it is also affected by writes to the SEEI and CEEI. These registers are provided so that the error interrupt enable for a single channel can easily be modified without the need to perform a read-modify-write sequence to the EEI register.

The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

Address: 2C0\_0000h base + 14h offset = 2C0\_0014h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EEI31	EEI30	EEI29	EEI28	EEI27	EEI26	EEI25	EEI24	EEI23	EEI22	EEI21	EEI20	EEI19	EEI18	EEI17	EEI16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	EEI15	EEI14	EEI13	EEI12	EEI11	EEI10	EEI9	EEI8	EEI7	EEI6	EEI5	EEI4	EEI3	EEI2	EEI1	EEI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DMA\_EEI field descriptions

Field	Description
0 EEI31	Enable Error Interrupt 31  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI30	Enable Error Interrupt 30  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
2 EEI29	Enable Error Interrupt 29  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI28	Enable Error Interrupt 28  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

Table continues on the next page...

**DMA\_EEI field descriptions (continued)**

Field	Description
4 EEI27	Enable Error Interrupt 27  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI26	Enable Error Interrupt 26  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI25	Enable Error Interrupt 25  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
7 EEI24	Enable Error Interrupt 24  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
8 EEI23	Enable Error Interrupt 23  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
9 EEI22	Enable Error Interrupt 22  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
10 EEI21	Enable Error Interrupt 21  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
11 EEI20	Enable Error Interrupt 20  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
12 EEI19	Enable Error Interrupt 19  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
13 EEI18	Enable Error Interrupt 18  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
14 EEI17	Enable Error Interrupt 17  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
15 EEI16	Enable Error Interrupt 16  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

*Table continues on the next page...*

**DMA\_EEI field descriptions (continued)**

Field	Description
16 EEI15	Enable Error Interrupt 15  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
17 EEI14	Enable Error Interrupt 14  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
18 EEI13	Enable Error Interrupt 13  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
19 EEI12	Enable Error Interrupt 12  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
20 EEI11	Enable Error Interrupt 11  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
21 EEI10	Enable Error Interrupt 10  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
22 EEI9	Enable Error Interrupt 9  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
23 EEI8	Enable Error Interrupt 8  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
24 EEI7	Enable Error Interrupt 7  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
25 EEI6	Enable Error Interrupt 6  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
26 EEI5	Enable Error Interrupt 5  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
27 EEI4	Enable Error Interrupt 4  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

*Table continues on the next page...*

**DMA\_EEI field descriptions (continued)**

Field	Description
28 EEI3	Enable Error Interrupt 3  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
29 EEI2	Enable Error Interrupt 2  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
30 EEI1	Enable Error Interrupt 1  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
31 EEI0	Enable Error Interrupt 0  0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

**21.4.10 Clear Enable Error Interrupt Register (DMA\_CEEI)**

The CEEI provides a simple memory-mapped mechanism to clear a given bit in the EEI to disable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be cleared. Setting the CAEE bit provides a global clear function, forcing the EEI contents to be cleared, disabling all DMA request inputs. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 18h offset = 2C0\_0018h

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	CAEE	0			CEEI		
Reset	0	0	0	0	0	0	0	0

**DMA\_CEEI field descriptions**

Field	Description
0 NOP	No Op enable  0 Normal operation 1 No operation, ignore the other bits in this register
1 CAEE	Clear All Enable Error Interrupts

*Table continues on the next page...*

**DMA\_CEEI field descriptions (continued)**

Field	Description
	0 Clear only the EEI bit specified in the CEEI field 1 Clear all bits in EEI
2 Reserved	This field is reserved.
3–7 CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

**21.4.11 Set Enable Error Interrupt Register (DMA\_SEEI)**

The SEEI provides a simple memory-mapped mechanism to set a given bit in the EEI to enable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be set. Setting the SAEE bit provides a global set function, forcing the entire EEI contents to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 19h offset = 2C0\_0019h

Bit	0	1	2	3	4	5	6	7
Read	0	0				0		
Write	NOP	SAEE	0			SEEI		
Reset	0	0	0	0	0	0	0	0

**DMA\_SEEI field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 SAEE	Sets All Enable Error Interrupts 0 Set only the EEI bit specified in the SEEI field. 1 Sets all bits in EEI
2 Reserved	This field is reserved.
3–7 SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

## 21.4.12 Clear Enable Request Register (DMA\_CERQ)

The CERQ provides a simple memory-mapped mechanism to clear a given bit in the ERQ to disable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be cleared. Setting the CAER bit provides a global clear function, forcing the entire contents of the ERQ to be cleared, disabling all DMA request inputs. If NOP is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

### NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

Address: 2C0\_0000h base + 1Ah offset = 2C0\_001Ah

Bit	0	1	2	3	4	5	6	7
Read	0	0				0		
Write	NOP	CAER	0			CERQ		
Reset	0	0	0	0	0	0	0	0

### DMA\_CERQ field descriptions

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 CAER	Clear All Enable Requests 0 Clear only the ERQ bit specified in the CERQ field 1 Clear all bits in ERQ
2 Reserved	This field is reserved.
3–7 CERQ	Clear Enable Request Clears the corresponding bit in ERQ.

### 21.4.13 Set Enable Request Register (DMA\_SERQ)

The SERQ provides a simple memory-mapped mechanism to set a given bit in the ERQ to enable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be set. Setting the SAER bit provides a global set function, forcing the entire contents of ERQ to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 1Bh offset = 2C0\_001Bh

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	SAER	0			SERQ		
Reset	0	0	0	0	0	0	0	0

**DMA\_SERQ field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 SAER	Set All Enable Requests 0 Set only the ERQ bit specified in the SERQ field 1 Set all bits in ERQ
2 Reserved	This field is reserved.
3–7 SERQ	Set Enable Request Sets the corresponding bit in ERQ.

## 21.4.14 Clear DONE Status Bit Register (DMA\_CDNE)

The CDNE provides a simple memory-mapped mechanism to clear the DONE bit in the TCD of the given channel. The data value on a register write causes the DONE bit in the corresponding transfer control descriptor to be cleared. Setting the CADN bit provides a global clear function, forcing all DONE bits to be cleared. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 1Ch offset = 2C0\_001Ch

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	CADN	0			CDNE		
Reset	0	0	0	0	0	0	0	0

**DMA\_CDNE field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 CADN	Clears All DONE Bits 0 Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1 Clears all bits in TCDn_CSR[DONE]
2 Reserved	This field is reserved.
3–7 CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

## 21.4.15 Set START Bit Register (DMA\_SSRT)

The SSRT provides a simple memory-mapped mechanism to set the START bit in the TCD of the given channel. The data value on a register write causes the START bit in the corresponding transfer control descriptor to be set. Setting the SAST bit provides a global set function, forcing all START bits to be set. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 1Dh offset = 2C0\_001Dh

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	SAST	0			SSRT		
Reset	0	0	0	0	0	0	0	0

**DMA\_SSRT field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 SAST	Set All START Bits (activates all channels) 0 Set only the TCDn_CSR[START] bit specified in the SSRT field 1 Set all bits in TCDn_CSR[START]
2 Reserved	This field is reserved.
3–7 SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

## 21.4.16 Clear Error Register (DMA\_CERR)

The CERR provides a simple memory-mapped mechanism to clear a given bit in the ERR to disable the error condition flag for a given channel. The given value on a register write causes the corresponding bit in the ERR to be cleared. Setting the CAEI bit provides a global clear function, forcing the ERR contents to be cleared, clearing all channel error indicators. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 1Eh offset = 2C0\_001Eh

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	CAEI	0			CERR		
Reset	0	0	0	0	0	0	0	0

**DMA\_CERR field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 CAEI	Clear All Error Indicators 0 Clear only the ERR bit specified in the CERR field 1 Clear all bits in ERR
2 Reserved	This field is reserved.
3–7 CERR	Clear Error Indicator Clears the corresponding bit in ERR

## 21.4.17 Clear Interrupt Request Register (DMA\_CINT)

The CINT provides a simple, memory-mapped mechanism to clear a given bit in the INT to disable the interrupt request for a given channel. The given value on a register write causes the corresponding bit in the INT to be cleared. Setting the CAIR bit provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: 2C0\_0000h base + 1Fh offset = 2C0\_001Fh

Bit	0	1	2	3	4	5	6	7
Read	0	0			0			
Write	NOP	CAIR	0			CINT		
Reset	0	0	0	0	0	0	0	0

**DMA\_CINT field descriptions**

Field	Description
0 NOP	No Op enable 0 Normal operation 1 No operation, ignore the other bits in this register
1 CAIR	Clear All Interrupt Requests 0 Clear only the INT bit specified in the CINT field 1 Clear all bits in INT
2 Reserved	This field is reserved.
3–7 CINT	Clear Interrupt Request Clears the corresponding bit in INT

## 21.4.18 Interrupt Request Register (DMA\_INT)

The INT register provides a bit map for the 32 channels signaling the presence of an interrupt request for each channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion. The outputs of this register are directly routed to the interrupt controller. During the interrupt-service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. Typically, a write to the CINT register in the interrupt service routine is used for this purpose.

The state of any given channel's interrupt request is directly affected by writes to this register; it is also affected by writes to the CINT register. On writes to INT, a 1 in any bit position clears the corresponding channel's interrupt request. A zero in any bit position has no affect on the corresponding channel's current interrupt status. The CINT register is provided so the interrupt request for a single channel can easily be cleared without the need to perform a read-modify-write sequence to the INT register.

Address: 2C0\_0000h base + 24h offset = 2C0\_0024h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	INT31	INT30	INT29	INT28	INT27	INT26	INT25	INT24	INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DMA\_INT field descriptions

Field	Description
0 INT31	Interrupt Request 31

Table continues on the next page...

**DMA\_INT field descriptions (continued)**

Field	Description
	0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
1 INT30	Interrupt Request 30 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
2 INT29	Interrupt Request 29 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
3 INT28	Interrupt Request 28 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
4 INT27	Interrupt Request 27 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
5 INT26	Interrupt Request 26 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
6 INT25	Interrupt Request 25 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
7 INT24	Interrupt Request 24 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
8 INT23	Interrupt Request 23 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
9 INT22	Interrupt Request 22 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
10 INT21	Interrupt Request 21 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
11 INT20	Interrupt Request 20 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
12 INT19	Interrupt Request 19

*Table continues on the next page...*

**DMA\_INT field descriptions (continued)**

Field	Description
	0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
13 INT18	Interrupt Request 18 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
14 INT17	Interrupt Request 17 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
15 INT16	Interrupt Request 16 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
16 INT15	Interrupt Request 15 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
17 INT14	Interrupt Request 14 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
18 INT13	Interrupt Request 13 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
19 INT12	Interrupt Request 12 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
20 INT11	Interrupt Request 11 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
21 INT10	Interrupt Request 10 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
22 INT9	Interrupt Request 9 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
23 INT8	Interrupt Request 8 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
24 INT7	Interrupt Request 7

*Table continues on the next page...*

**DMA\_INT field descriptions (continued)**

Field	Description
	0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
25 INT6	Interrupt Request 6 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
26 INT5	Interrupt Request 5 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
27 INT4	Interrupt Request 4 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
28 INT3	Interrupt Request 3 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
29 INT2	Interrupt Request 2 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
30 INT1	Interrupt Request 1 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
31 INT0	Interrupt Request 0 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

**21.4.19 Error Register (DMA\_ERR)**

The ERR provides a bit map for the 32 channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI, then logically summed across groups of 16 and 32 channels to form several group error interrupt requests, which are then routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. Typically, a write to the CERR in the interrupt-service routine is used for this purpose. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected.

The contents of this register can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI. The state of any given channel's error indicators is affected by writes to this register; it is also affected by writes to the CERR. On writes to the ERR, a one in any bit position clears the corresponding channel's error status. A zero in any bit position has no affect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be cleared.

Address: 2C0\_0000h base + 2Ch offset = 2C0\_002Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ERR31	ERR30	ERR29	ERR28	ERR27	ERR26	ERR25	ERR24	ERR23	ERR22	ERR21	ERR20	ERR19	ERR18	ERR17	ERR16
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ERR15	ERR14	ERR13	ERR12	ERR11	ERR10	ERR9	ERR8	ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1	ERR0
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DMA\_ERR field descriptions

Field	Description
0 ERR31	Error In Channel 31  0 An error in this channel has not occurred 1 An error in this channel has occurred
1 ERR30	Error In Channel 30  0 An error in this channel has not occurred 1 An error in this channel has occurred
2 ERR29	Error In Channel 29  0 An error in this channel has not occurred 1 An error in this channel has occurred
3 ERR28	Error In Channel 28  0 An error in this channel has not occurred 1 An error in this channel has occurred

Table continues on the next page...

**DMA\_ERR field descriptions (continued)**

Field	Description
4 ERR27	Error In Channel 27  0 An error in this channel has not occurred 1 An error in this channel has occurred
5 ERR26	Error In Channel 26  0 An error in this channel has not occurred 1 An error in this channel has occurred
6 ERR25	Error In Channel 25  0 An error in this channel has not occurred 1 An error in this channel has occurred
7 ERR24	Error In Channel 24  0 An error in this channel has not occurred 1 An error in this channel has occurred
8 ERR23	Error In Channel 23  0 An error in this channel has not occurred 1 An error in this channel has occurred
9 ERR22	Error In Channel 22  0 An error in this channel has not occurred 1 An error in this channel has occurred
10 ERR21	Error In Channel 21  0 An error in this channel has not occurred 1 An error in this channel has occurred
11 ERR20	Error In Channel 20  0 An error in this channel has not occurred 1 An error in this channel has occurred
12 ERR19	Error In Channel 19  0 An error in this channel has not occurred 1 An error in this channel has occurred
13 ERR18	Error In Channel 18  0 An error in this channel has not occurred 1 An error in this channel has occurred
14 ERR17	Error In Channel 17  0 An error in this channel has not occurred 1 An error in this channel has occurred
15 ERR16	Error In Channel 16  0 An error in this channel has not occurred 1 An error in this channel has occurred

*Table continues on the next page...*

**DMA\_ERR field descriptions (continued)**

Field	Description
16 ERR15	Error In Channel 15  0 An error in this channel has not occurred 1 An error in this channel has occurred
17 ERR14	Error In Channel 14  0 An error in this channel has not occurred 1 An error in this channel has occurred
18 ERR13	Error In Channel 13  0 An error in this channel has not occurred 1 An error in this channel has occurred
19 ERR12	Error In Channel 12  0 An error in this channel has not occurred 1 An error in this channel has occurred
20 ERR11	Error In Channel 11  0 An error in this channel has not occurred 1 An error in this channel has occurred
21 ERR10	Error In Channel 10  0 An error in this channel has not occurred 1 An error in this channel has occurred
22 ERR9	Error In Channel 9  0 An error in this channel has not occurred 1 An error in this channel has occurred
23 ERR8	Error In Channel 8  0 An error in this channel has not occurred 1 An error in this channel has occurred
24 ERR7	Error In Channel 7  0 An error in this channel has not occurred 1 An error in this channel has occurred
25 ERR6	Error In Channel 6  0 An error in this channel has not occurred 1 An error in this channel has occurred
26 ERR5	Error In Channel 5  0 An error in this channel has not occurred 1 An error in this channel has occurred
27 ERR4	Error In Channel 4  0 An error in this channel has not occurred 1 An error in this channel has occurred

*Table continues on the next page...*

**DMA\_ERR field descriptions (continued)**

Field	Description
28 ERR3	Error In Channel 3  0 An error in this channel has not occurred 1 An error in this channel has occurred
29 ERR2	Error In Channel 2  0 An error in this channel has not occurred 1 An error in this channel has occurred
30 ERR1	Error In Channel 1  0 An error in this channel has not occurred 1 An error in this channel has occurred
31 ERR0	Error In Channel 0  0 An error in this channel has not occurred 1 An error in this channel has occurred

**21.4.20 Hardware Request Status Register (DMA\_HRS)**

The HRS register provides a bit map for the DMA channels, signaling the presence of a hardware request for each channel. The hardware request status bits reflect the current state of the register and qualified (via the ERQ fields) DMA request signals as seen by the DMA's arbitration logic. This view into the hardware request signals may be used for debug purposes.

**NOTE**

These bits reflect the state of the request as seen by the arbitration logic. Therefore, this status is affected by the ERQ bits.

Address: 2C0\_0000h base + 34h offset = 2C0\_0034h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	HRS31	HRS30	HRS29	HRS28	HRS27	HRS26	HRS25	HRS24	HRS23	HRS22	HRS21	HRS20	HRS19	HRS18	HRS17	HRS16
W																

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	HRS15	HRS14	HRS13	HRS12	HRS11	HRS10	HRS9	HRS8	HRS7	HRS6	HRS5	HRS4	HRS3	HRS2	HRS1	HRS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DMA\_HRS field descriptions**

Field	Description
0 HRS31	<p>Hardware Request Status Channel 31</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 31 is not present 1 A hardware service request for channel 31 is present</p>
1 HRS30	<p>Hardware Request Status Channel 30</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 30 is not present 1 A hardware service request for channel 30 is present</p>
2 HRS29	<p>Hardware Request Status Channel 29</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 29 is not preset 1 A hardware service request for channel 29 is present</p>
3 HRS28	<p>Hardware Request Status Channel 28</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 28 is not present 1 A hardware service request for channel 28 is present</p>
4 HRS27	<p>Hardware Request Status Channel 27</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p>

*Table continues on the next page...*

**DMA\_HRS field descriptions (continued)**

Field	Description
	<p>0 A hardware service request for channel 27 is not present 1 A hardware service request for channel 27 is present</p>
5 HRS26	<p>Hardware Request Status Channel 26</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 26 is not present 1 A hardware service request for channel 26 is present</p>
6 HRS25	<p>Hardware Request Status Channel 25</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 25 is not present 1 A hardware service request for channel 25 is present</p>
7 HRS24	<p>Hardware Request Status Channel 24</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 24 is not present 1 A hardware service request for channel 24 is present</p>
8 HRS23	<p>Hardware Request Status Channel 23</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 23 is not present 1 A hardware service request for channel 23 is present</p>
9 HRS22	<p>Hardware Request Status Channel 22</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 22 is not present 1 A hardware service request for channel 22 is present</p>
10 HRS21	<p>Hardware Request Status Channel 21</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 21 is not present 1 A hardware service request for channel 21 is present</p>
11 HRS20	Hardware Request Status Channel 20

*Table continues on the next page...*

**DMA\_HRS field descriptions (continued)**

Field	Description
	<p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 20 is not present 1 A hardware service request for channel 20 is present</p>
12 HRS19	<p>Hardware Request Status Channel 19</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 19 is not present 1 A hardware service request for channel 19 is present</p>
13 HRS18	<p>Hardware Request Status Channel 18</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 18 is not present 1 A hardware service request for channel 18 is present</p>
14 HRS17	<p>Hardware Request Status Channel 17</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 17 is not present 1 A hardware service request for channel 17 is present</p>
15 HRS16	<p>Hardware Request Status Channel 16</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 16 is not present 1 A hardware service request for channel 16 is present</p>
16 HRS15	<p>Hardware Request Status Channel 15</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 15 is not present 1 A hardware service request for channel 15 is present</p>
17 HRS14	<p>Hardware Request Status Channel 14</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p>

*Table continues on the next page...*

**DMA\_HRS field descriptions (continued)**

Field	Description
	<p>0 A hardware service request for channel 14 is not present 1 A hardware service request for channel 14 is present</p>
18 HRS13	<p>Hardware Request Status Channel 13</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 13 is not present 1 A hardware service request for channel 13 is present</p>
19 HRS12	<p>Hardware Request Status Channel 12</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 12 is not present 1 A hardware service request for channel 12 is present</p>
20 HRS11	<p>Hardware Request Status Channel 11</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 11 is not present 1 A hardware service request for channel 11 is present</p>
21 HRS10	<p>Hardware Request Status Channel 10</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 10 is not present 1 A hardware service request for channel 10 is present</p>
22 HRS9	<p>Hardware Request Status Channel 9</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 9 is not present 1 A hardware service request for channel 9 is present</p>
23 HRS8	<p>Hardware Request Status Channel 8</p> <p>The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.</p> <p>0 A hardware service request for channel 8 is not present 1 A hardware service request for channel 8 is present</p>
24 HRS7	Hardware Request Status Channel 7

*Table continues on the next page...*

**DMA\_HRS field descriptions (continued)**

Field	Description
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 7 is not present 1 A hardware service request for channel 7 is present
25 HRS6	Hardware Request Status Channel 6  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 6 is not present 1 A hardware service request for channel 6 is present
26 HRS5	Hardware Request Status Channel 5  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 5 is not present 1 A hardware service request for channel 5 is present
27 HRS4	Hardware Request Status Channel 4  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 4 is not present 1 A hardware service request for channel 4 is present
28 HRS3	Hardware Request Status Channel 3  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 3 is not present 1 A hardware service request for channel 3 is present
29 HRS2	Hardware Request Status Channel 2  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 2 is not present 1 A hardware service request for channel 2 is present
30 HRS1	Hardware Request Status Channel 1  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.

*Table continues on the next page...*

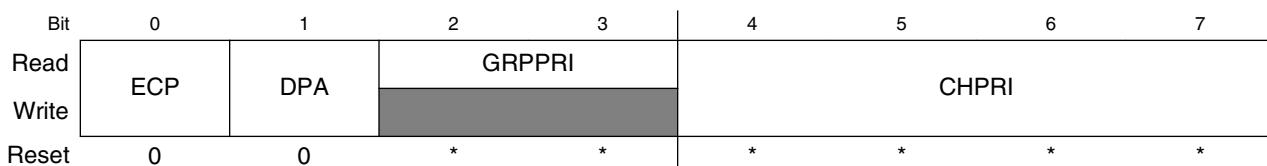
**DMA\_HRS field descriptions (continued)**

Field	Description
	0 A hardware service request for channel 1 is not present 1 A hardware service request for channel 1 is present
31 HRS0	Hardware Request Status Channel 0  The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware.  0 A hardware service request for channel 0 is not present 1 A hardware service request for channel 0 is present

**21.4.21 Channel n Priority Register (DMA\_DCHPRI<sub>n</sub>)**

When fixed-priority channel arbitration is enabled (CR[ERCA] = 0), the contents of these registers define the unique priorities associated with each channel within a group. The channel priorities are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, a configuration error is reported. The range of the priority value is limited to the values of 0 through 15. When read, the GRPPRI bits of the DCHPRI<sub>n</sub> register reflect the current priority level of the group of channels in which the corresponding channel resides. GRPPRI bits are not affected by writes to the DCHPRI<sub>n</sub> registers. The group priority is assigned in the DMA control register.

Address: 2C0\_0000h base + 100h offset + (1d × i), where i=0d to 31d



\* Notes:

- CHPRI field: See bit field description.
- GRPPRI field: See bit field description.

**DMA\_DCHPRI<sub>n</sub> field descriptions**

Field	Description
0 ECP	Enable Channel Preemption.  0 Channel n cannot be suspended by a higher priority channel's service request. 1 Channel n can be temporarily suspended by the service request of a higher priority channel.

*Table continues on the next page...*

**DMA\_DCHPRI<sub>n</sub> field descriptions (continued)**

Field	Description
1 DPA	Disable Preempt Ability. 0 Channel n can suspend a lower priority channel. 1 Channel n cannot suspend any channel, regardless of channel priority.
2–3 GRPPRI	Channel n Current Group Priority Group priority assigned to this channel group when fixed-priority arbitration is enabled. This field is read-only; writes are ignored. <b>NOTE:</b> Reset value for the group and channel priority fields, GRPPRI and CHPRI, is equal to the corresponding channel number for each priority register, that is, DCHPRI31[GRPPRI] = 0b01 and DCHPRI31[CHPRI] equals 0b1111.
4–7 CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled <b>NOTE:</b> Reset value for the group and channel priority fields, GRPPRI and CHPRI, is equal to the corresponding channel number for each priority register, that is, DCHPRI31[GRPPRI] = 0b01 and DCHPRI31[CHPRI] = 0b01111.

**21.4.22 TCD Source Address (DMA\_TCD<sub>n</sub>\_SADDR)**

Address: 2C0\_0000h base + 1000h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	SADDR															
W																																

Reset x\* x\*

\* Notes:

- x = Undefined at reset.

**DMA\_TCD<sub>n</sub>\_SADDR field descriptions**

Field	Description
0–31 SADDR	Source Address Memory address pointing to the source data.

## 21.4.23 TCD Signed Source Address Offset (DMA\_TCDn\_SOFF)

Address: 2C0\_0000h base + 1004h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*
Write	SOFF															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_SOFF field descriptions

Field	Description															
0–15 SOFF	Source address signed offset  Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.															

## 21.4.24 TCD Transfer Attributes (DMA\_TCDn\_ATTR)

Address: 2C0\_0000h base + 1006h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read	SMOD				SSIZE				DMOD				DSIZE			
Write	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_ATTR field descriptions

Field	Description															
0–4 SMOD	Source Address Modulo  0    Source address modulo feature is disabled ≠0   This value defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range.															
5–7 SSIZE	Source data transfer size  <b>NOTE:</b> Using a Reserved value causes a configuration error.															

Table continues on the next page...

**DMA\_TCDn\_ATTR field descriptions (continued)**

Field	Description
	The eDMA defaults to privileged data access for all transactions.  000 8-bit 001 16-bit 010 32-bit 011 64-bit 100 Reserved 101 32-byte burst (4 beats of 64 bits) 110 Reserved 111 Reserved
8–12 DMOD	Destination Address Modulo  See the SMOD definition
13–15 DSIZE	Destination data transfer size  See the SSIZE definition

### 21.4.25 TCD Minor Byte Count (Minor Loop Mapping Disabled) (DMA\_TCDn\_NBYTES\_MLNO)

This register, or one of the next two registers (TCD\_NBYTES\_Mloffno, TCD\_NBYTES\_Mloffyes), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is disabled (CR[EMLM] = 0)

If minor loop mapping is enabled, see the TCD\_NBYTES\_Mloffno and TCD\_NBYTES\_Mloffyes register descriptions for the definition of TCD word 2.

Address: 2C0\_0000h base + 1008h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset x\* | x\* x\*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_NBYTES\_MLNO field descriptions**

Field	Description
0–31 NBYTES	Minor Byte Transfer Count

**DMA\_TCDn\_NBYTES\_MLNO field descriptions (continued)**

Field	Description
	<p>Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.</p> <p><b>NOTE:</b> An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.</p>

### 21.4.26 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (DMA\_TCDn\_NBYTES\_MLOFFNO)

One of three registers (this register, TCD\_NBYTES\_MLNO, or TCD\_NBYTES\_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- SMLOE = 0 and DMLOE = 0

If minor loop mapping is enabled and SMLOE or DMLOE is set, then refer to the TCD\_NBYTES\_MLOFFYES register description. If minor loop mapping is disabled, then refer to the TCD\_NBYTES\_MLNO register description.

Address: 2C0\_0000h base + 1008h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SMLOE	DMLOE														
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_NBYTES\_Mloffno field descriptions**

Field	Description
0 SMLOE	Source Minor Loop Offset Enable  Selects whether the minor loop offset is applied to the source address upon minor loop completion.  0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
1 DMLOE	Destination Minor Loop Offset enable  Selects whether the minor loop offset is applied to the destination address upon minor loop completion.  0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
2–31 NBYTES	Minor Byte Transfer Count  Number of bytes to be transferred in each service request of the channel.  As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

**21.4.27 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (DMA\_TCDn\_NBYTES\_Mloffyes)**

One of three registers (this register, TCD\_NBYTES\_MLNO, or TCD\_NBYTES\_Mloffno), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset is enabled (SMLOE or DMLOE = 1)

If minor loop mapping is enabled and SMLOE and DMLOE are cleared, then refer to the TCD\_NBYTES\_Mloffno register description. If minor loop mapping is disabled, then refer to the TCD\_NBYTES\_MLNO register description.

## Memory map/register definition

Address: 2C0\_0000h base + 1008h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SMLOE	DMLOE														Mloff
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																Nbytes
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

## DMA\_TCDn\_NBYTES\_MloffYES field descriptions

Field	Description
0 SMLOE	Source Minor Loop Offset Enable  Selects whether the minor loop offset is applied to the source address upon minor loop completion.  0 The minor loop offset is not applied to the SADDR 1 The minor loop offset is applied to the SADDR
1 DMLOE	Destination Minor Loop Offset enable  Selects whether the minor loop offset is applied to the destination address upon minor loop completion.  0 The minor loop offset is not applied to the DADDR 1 The minor loop offset is applied to the DADDR
2–21 Mloff	If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
22–31 Nbytes	Minor Byte Transfer Count  Number of bytes to be transferred in each service request of the channel.  As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

## 21.4.28 TCD Last Source Address Adjustment (DMA\_TCDn\_SLAST)

Address: 2C0\_0000h base + 100Ch offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset x\* x\*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_SLAST field descriptions

Field	Description
0–31 SLAST	Last Source Address Adjustment  Adjustment value added to the source address at the completion of the major iteration count. This value can be applied to restore the source address to the initial value, or adjust the address to reference the next data structure.  This register uses two's complement notation; the overflow bit is discarded.

## 21.4.29 TCD Destination Address (DMA\_TCDn\_DADDR)

Address: 2C0\_0000h base + 1010h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset x\* x\*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_DADDR field descriptions

Field	Description
0–31 DADDR	Destination Address  Memory address pointing to the destination data.

## 21.4.30 TCD Signed Destination Address Offset (DMA\_TCDn\_DOFF)

Address: 2C0\_0000h base + 1014h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read	DOFF															
Write	DOFF															
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_DOFF field descriptions

Field	Description
0–15 DOFF	Destination Address Signed Offset  Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

## 21.4.31 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA\_TCDn\_CITER\_ELINKYES)

If TCDn\_CITER[ELINK] is set, the TCDn\_CITER register is defined as follows.

Address: 2C0\_0000h base + 1016h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7
Read	ELINK							
Write	0							
Reset	x*	x*	x*	x*	x*	x*	x*	x*
Bit	8	9	10	11	12	13	14	15
Read	CITER							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

### DMA\_TCDn\_CITER\_ELINKYES field descriptions

Field	Description
0 ELINK	Enable channel-to-channel linking on minor-loop complete  As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel.

Table continues on the next page...

**DMA\_TCDn\_CITER\_ELINKYES field descriptions (continued)**

Field	Description
	If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.  <b>NOTE:</b> This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported.  0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
1 Reserved	This field is reserved.
2–6 LINKCH	Minor Loop Link Channel Number  If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7–15 CITER	Current Major Iteration Count  This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.  <b>NOTE:</b> When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field.  <b>NOTE:</b> If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

### 21.4.32 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA\_TCDn\_CITER\_ELINKNO)

If TCDn\_CITER[ELINK] is cleared, the TCDn\_CITER register is defined as follows.

Address: 2C0\_0000h base + 1016h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7
Read Write	ELINK							
Reset	x*	x*	x*	x*	x*	x*	x*	x*
Bit	8	9	10	11	12	13	14	15
Read Write	CITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_CITER\_ELINKNO field descriptions**

Field	Description
0 ELINK	<p>Enable channel-to-channel linking on minor-loop complete</p> <p>As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel.</p> <p>If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.</p> <p><b>NOTE:</b> This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported.</p> <p>0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled</p>
1–15 CITER	<p>Current Major Iteration Count</p> <p>This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.</p> <p><b>NOTE:</b> When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field.</p> <p><b>NOTE:</b> If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.</p>

**21.4.33 TCD Last Destination Address Adjustment/Scatter Gather Address (DMA\_TCDn\_DLASTSGA)**

Address: 2C0\_0000h base + 1018h offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset x\* x\*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_DLASTSGA field descriptions**

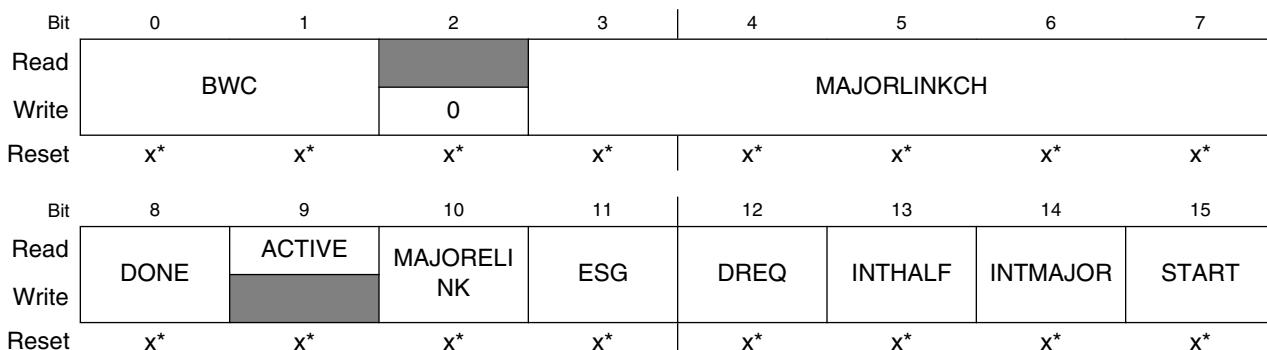
Field	Description
0–31 DLASTSGA	<p>Destination last address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather).</p> <p>If (TCDn_CSR[ESG] = 0) then:</p> <ul style="list-style-type: none"> <li>• Adjustment value added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure.</li> <li>• This field uses two's complement notation for the final destination address adjustment.</li> </ul>

**DMA\_TCDn\_DLASTSGA field descriptions (continued)**

Field	Description
	<p>Otherwise:</p> <ul style="list-style-type: none"> <li>This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.</li> </ul>

**21.4.34 TCD Control and Status (DMA\_TCDn\_CSR)**

Address: 2C0\_0000h base + 101Ch offset + (32d × i), where i=0d to 31d



\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_CSR field descriptions**

Field	Description
0–1 BWC	<p>Bandwidth Control</p> <p>Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces the eDMA to stall after the completion of each read/write access to control the bus request bandwidth seen by the crossbar switch.</p> <p><b>NOTE:</b> If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency.</p> <p>00 No eDMA engine stalls. 10 eDMA engine stalls for 4 cycles after each R/W. 11 eDMA engine stalls for 8 cycles after each R/W.</p>
2 Reserved	This field is reserved.
3–7 MAJORLINKCH	<p>Major Loop Link Channel Number</p> <p>If (MAJORELINK = 0) then:</p> <ul style="list-style-type: none"> <li>No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted.</li> </ul>

*Table continues on the next page...*

**DMA\_TCDn\_CSR field descriptions (continued)**

Field	Description
	<p>Otherwise:</p> <ul style="list-style-type: none"> <li>After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.</li> </ul>
8 DONE	<p>Channel Done</p> <p>This flag indicates the eDMA has completed the major loop. The eDMA engine sets it as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated.</p> <p>The access of this field is W0C, write-zero-to-clear.</p> <p><b>NOTE:</b> This bit must be cleared to write the MAJORELINK or ESG bits.</p>
9 ACTIVE	<p>Channel Active</p> <p>This flag signals the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.</p>
10 MAJORELINK	<p>Enable channel-to-channel linking on major loop complete</p> <p>As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel.</p> <p><b>NOTE:</b> To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set.</p> <p>0 The channel-to-channel linking is disabled. 1 The channel-to-channel linking is enabled.</p>
11 ESG	<p>Enable Scatter/Gather Processing</p> <p>As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory.</p> <p><b>NOTE:</b> To support the dynamic scatter/gather coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set.</p> <p>0 The current channel's TCD is normal format. 1 The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.</p>
12 DREQ	<p>Disable Request</p> <p>If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero.</p> <p>0 The channel's ERQ bit is not affected. 1 The channel's ERQ bit is cleared when the major loop is complete.</p>
13 INTHALF	<p>Enable an interrupt when major counter is half complete.</p> <p>If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER &gt;&gt; 1)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress.</p> <p><b>NOTE:</b> If BITER = 1, do not use INTHALF. Use INTMAJOR instead.</p>

*Table continues on the next page...*

**DMA\_TCDn\_CSR field descriptions (continued)**

Field	Description
	<p>0 The half-point interrupt is disabled. 1 The half-point interrupt is enabled.</p>
14 INTMAJOR	<p>Enable an interrupt when major iteration count completes.  If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count reaches zero.</p> <p>0 The end-of-major loop interrupt is disabled. 1 The end-of-major loop interrupt is enabled.</p>
15 START	<p>Channel Start  If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution.</p> <p>0 The channel is not explicitly started. 1 The channel is explicitly started via a software initiated service request.</p>

**21.4.35 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA\_TCDn\_BITER\_ELINKYES)**

If the TCDn\_BITER[ELINK] bit is set, the TCDn\_BITER register is defined as follows.

Address: 2C0\_0000h base + 101Eh offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7
Read	ELINK				LINKCH			BITER
Write		0						
Reset	x*	x*	x*	x*	x*	x*	x*	x*
Bit	8	9	10	11	12	13	14	15
Read				BITER				
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_BITER\_ELINKYES field descriptions**

Field	Description
0 ELINK	<p>Enables channel-to-channel linking on minor loop complete  As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.</p>

*Table continues on the next page...*

**DMA\_TCDn\_BITER\_ELINKYES field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.</p> <p>0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled</p>
1 Reserved	This field is reserved.
2–6 LINKCH	<p>Link Channel Number</p> <p>If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.</p> <p><b>NOTE:</b> When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.</p>
7–15 BITER	<p>Starting major iteration count</p> <p>As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.</p> <p><b>NOTE:</b> When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.</p>

### 21.4.36 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA\_TCDn\_BITER\_ELINKNO)

If the TCDn\_BITER[ELINK] bit is cleared, the TCDn\_BITER register is defined as follows.

Address: 2C0\_0000h base + 101Eh offset + (32d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7
Read Write	ELINK							
Reset	x*	x*	x*	x*	x*	x*	x*	x*
Bit	8	9	10	11	12	13	14	15
Read Write	BITER							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

\* Notes:

- x = Undefined at reset.

**DMA\_TCDn\_BITER\_ELINKNO field descriptions**

Field	Description
0 ELINK	<p>Enables channel-to-channel linking on minor loop complete</p> <p>As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking.</p> <p><b>NOTE:</b> When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.</p> <ul style="list-style-type: none"> <li>0 The channel-to-channel linking is disabled</li> <li>1 The channel-to-channel linking is enabled</li> </ul>
1–15 BITER	<p>Starting Major Iteration Count</p> <p>As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.</p> <p><b>NOTE:</b> When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.</p>

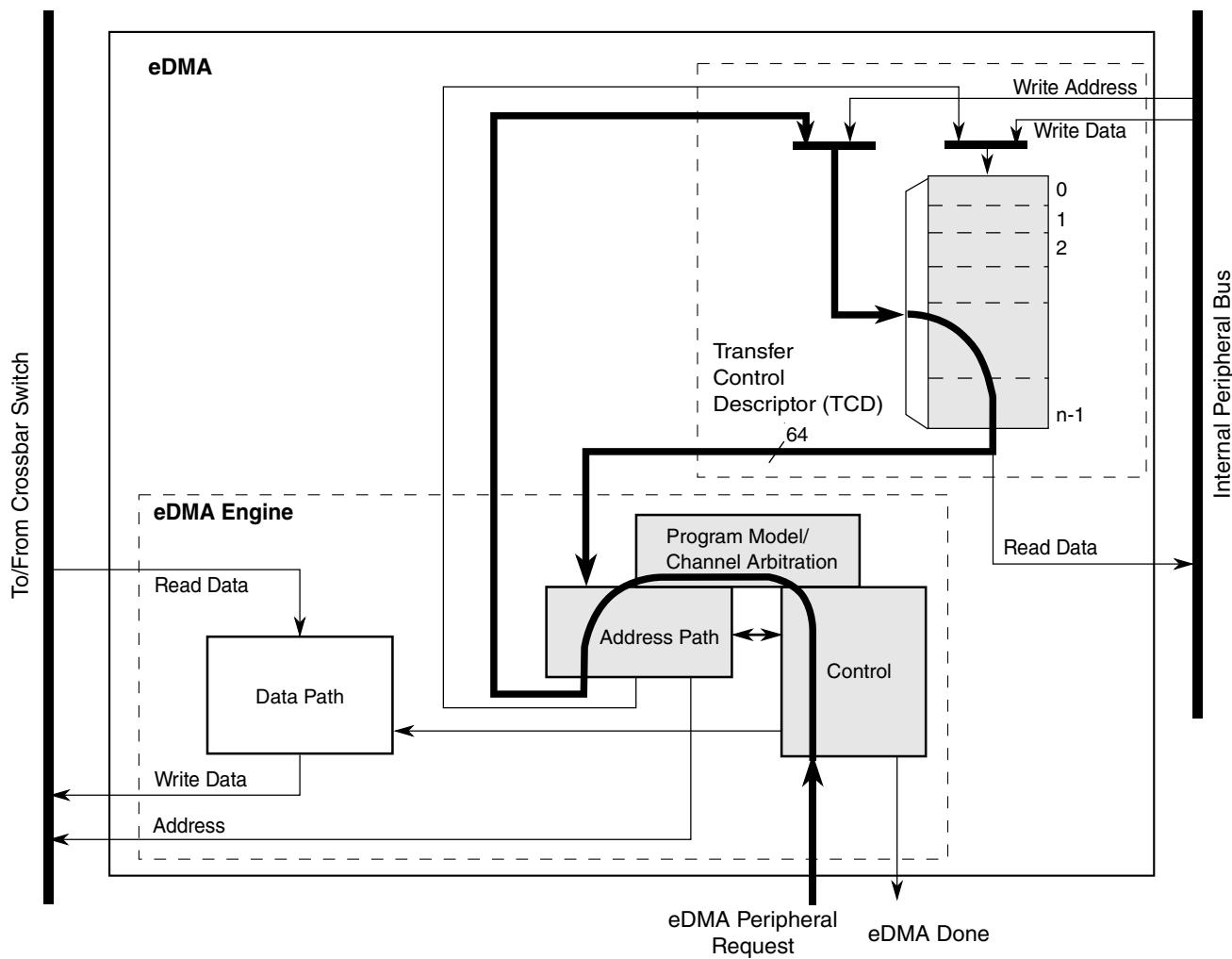
## 21.5 Functional description

The operation of the eDMA is described in the following subsections.

### 21.5.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:



**Figure 21-3. eDMA operation, part 1**

This example uses the assertion of the eDMA peripheral request signal to request service for channel  $n$ . Channel activation via software and the TCD<sub>n</sub>\_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration. In the next cycle, the channel arbitration performs, using the fixed-priority or round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD<sub>n</sub>. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path channel x or y registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path channel x or y registers.

The following diagram illustrates the second part of the basic data flow:

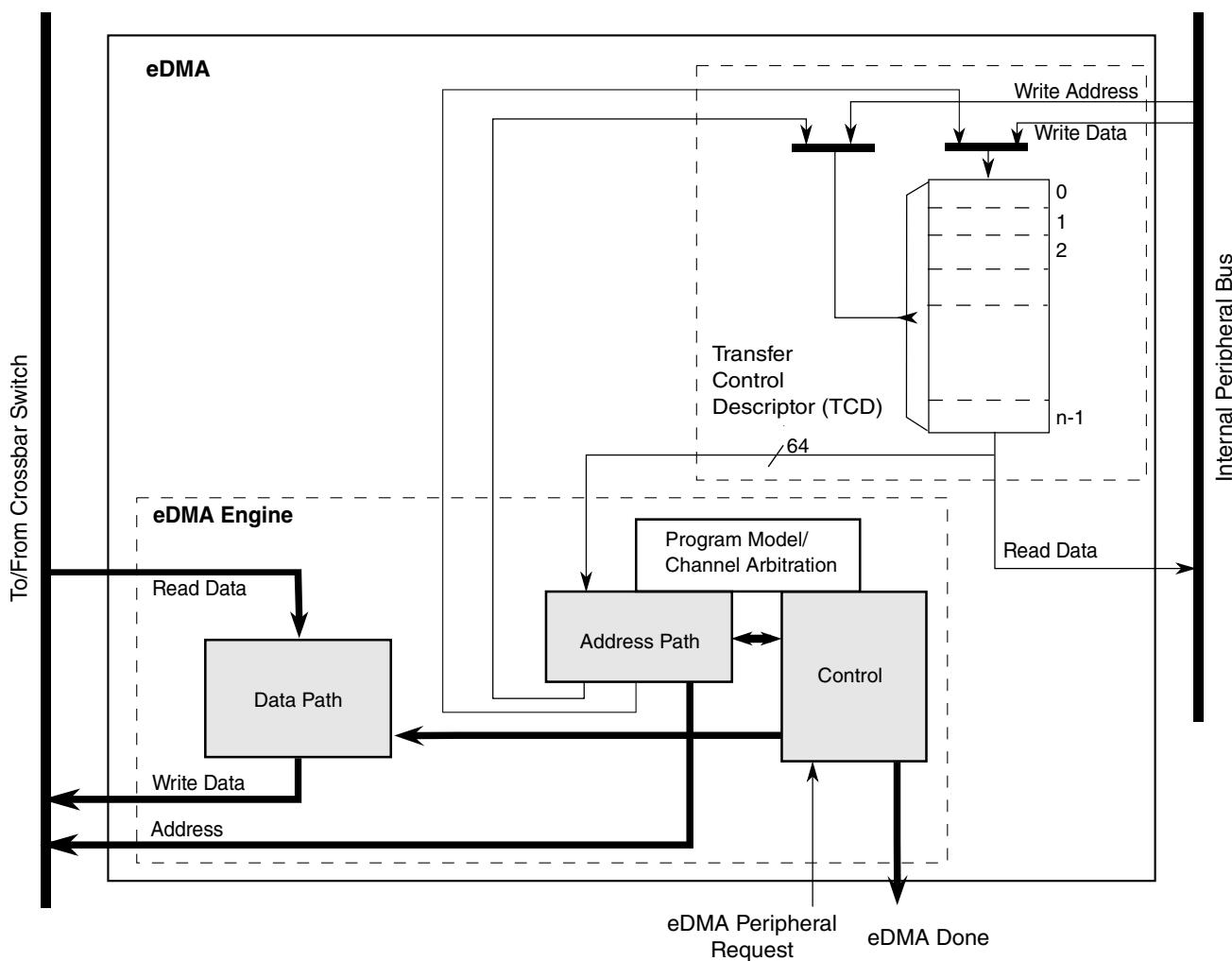


Figure 21-4. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the minor byte count has transferred.

After the minor byte count has moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

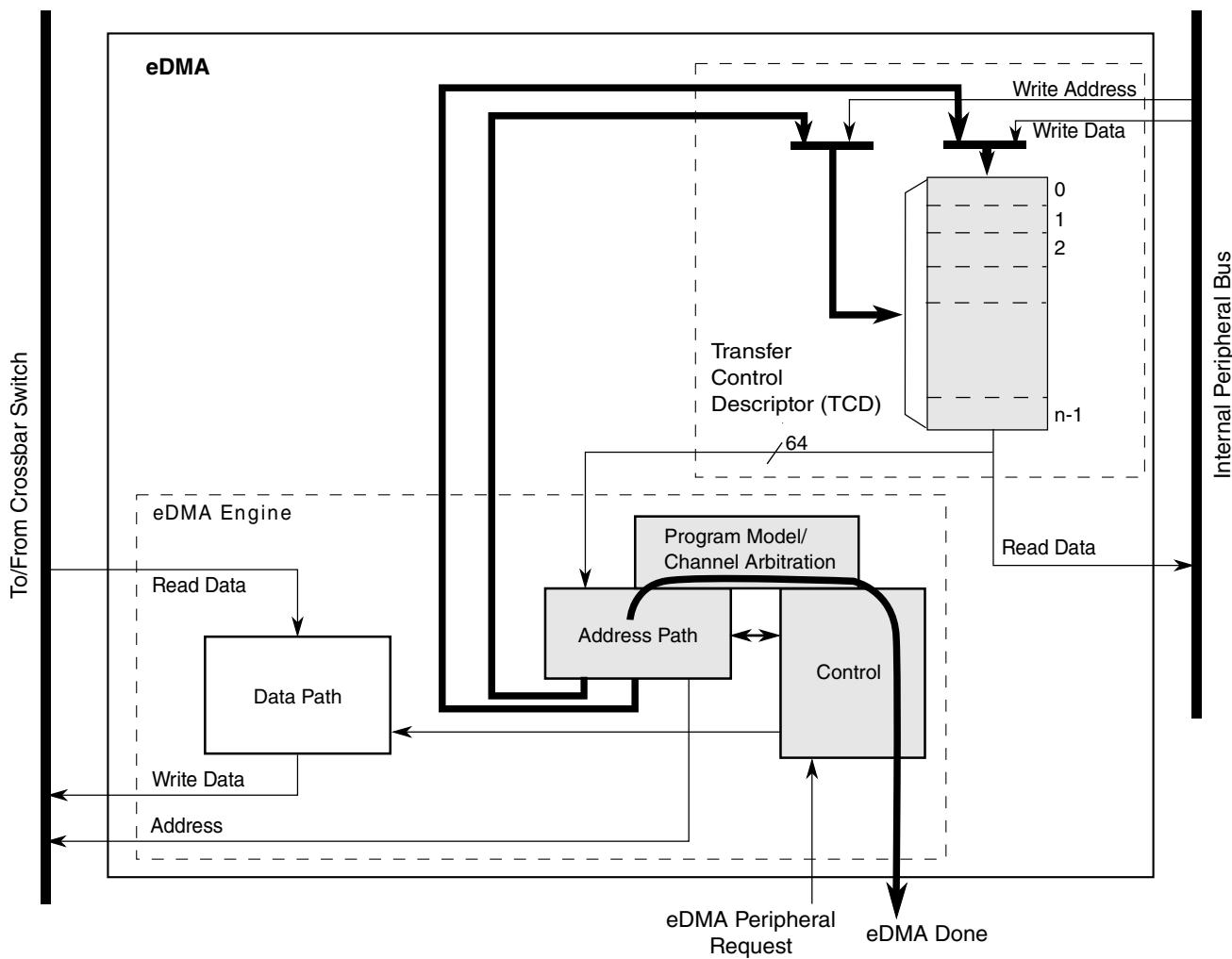


Figure 21-5. eDMA operation, part 3

## 21.5.2 Fault reporting and handling

Channel errors are reported in the Error Status register (DMAx\_ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor or an illegal priority register setting in Fixed-Arbitration mode, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.

- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.
- In fixed arbitration mode, a configuration error is caused by any two channel priorities being equal. All channel priority levels must be unique when fixed arbitration mode is enabled.

## NOTE

When two channels have the same priority, a channel priority error exists and will be reported in the Error Status register. However, the channel number will not be reported in the Error Status register. When all of the channel priorities within a group are not unique, the channel number selected by arbitration is undetermined.

To aid in Channel Priority Error (CPE) debug, set the Halt On Error bit in the DMA's Control Register. If all of the channel priorities within a group are not unique, the DMA will be halted after the CPE error is recorded. The DMA will remain halted and will not process any channel service requests. Once all of the channel priorities are set to unique numbers, the DMA may be enabled again by clearing the Halt bit.

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST\_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn\_CITER[E\_LINK] bit does not equal the TCDn\_BITER[E\_LINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error

occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

A transfer may be cancelled by software with the CR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. The current read-write sequence is allowed to finish. If the cancel occurs on the last read-write sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (DMAx\_ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into DMA\_ES[ERRCHN] and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

### **NOTE**

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed.

The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The application software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (DMAx\_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

### 21.5.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the DCHPRIn[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted. Preemption is available only when fixed arbitration is selected.

A channel's ability to preempt another channel can be disabled by setting DCHPRIn[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel.

## 21.6 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

### 21.6.1 eDMA initialization

To initialize the eDMA:

1. Write to the CR if a configuration other than the default is desired.
2. Write the channel priority levels to the DCHPRIn registers if a configuration other than the default is desired.
3. Enable error interrupts in the EEI register if so desired.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the ERQ register.

6. Request channel service via either:

- Software: setting the `TCDn_CSR[START]`
- Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

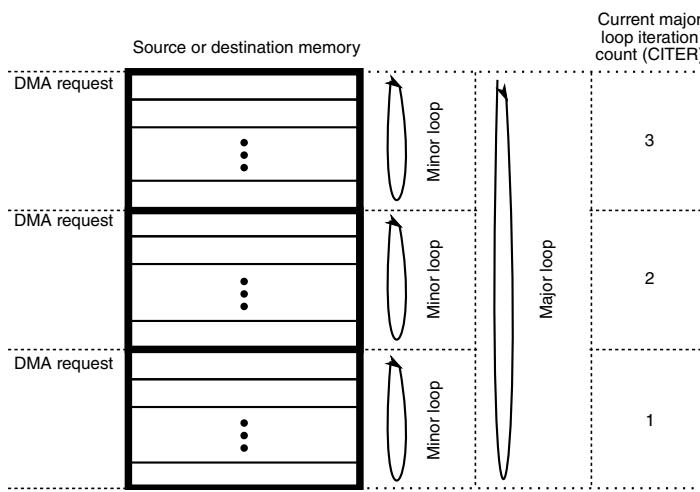
As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by `TCDn_SADDR`, to the destination, as defined by `TCDn_DADDR`, continue until the number of bytes specified by `TCDn_NBYTES` are transferred.

When the transfer is complete, the eDMA engine's local `TCDn_SADDR`, `TCDn_DADDR`, and `TCDn_CITER` are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

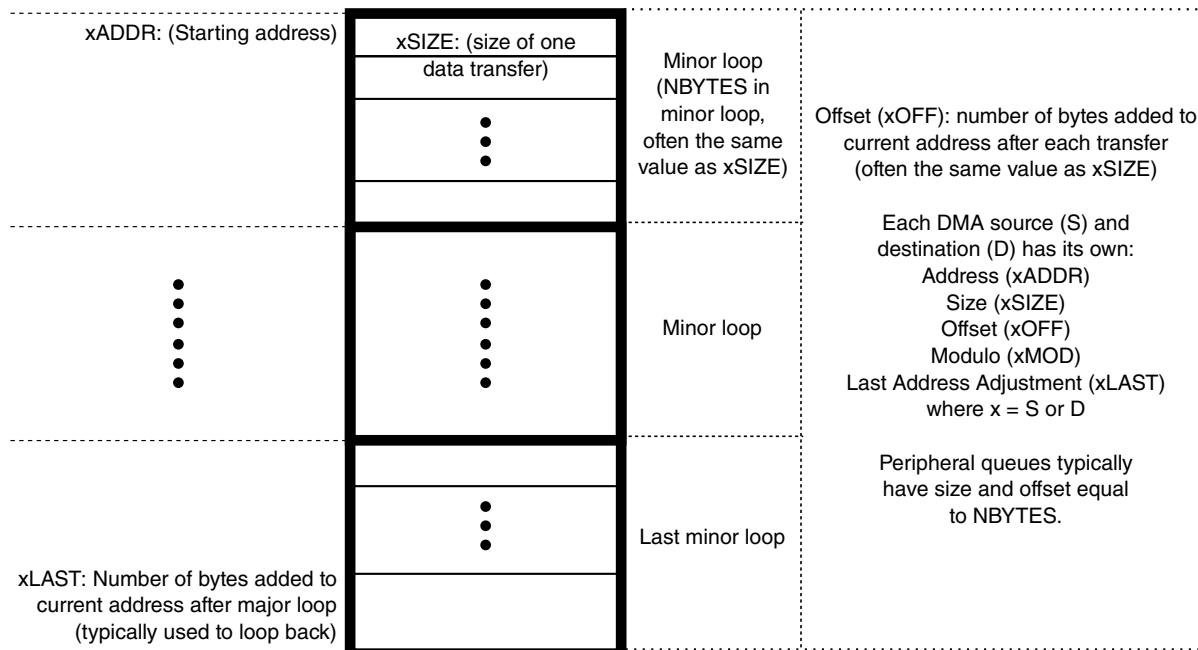
**Table 21-4. TCD Control and Status fields**

<code>TCDn_CSR</code> field name	Description
<code>START</code>	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
<code>ACTIVE</code>	Status bit indicating the channel is currently in execution
<code>DONE</code>	Status bit indicating major loop completion (cleared by software when using a software initiated DMA service)
<code>D_REQ</code>	Control bit to disable DMA request at end of major loop completion when using a hardware initiated DMA service
<code>BWC</code>	Control bits for throttling bandwidth control of a channel
<code>E_SG</code>	Control bit to enable scatter-gather feature
<code>INT_HALF</code>	Control bit to enable interrupt when major loop is half complete
<code>INT_MAJ</code>	Control bit to enable interrupt when major loop completes

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

**Figure 21-6. Example of multiple loop iterations**

The following figure lists the memory array terms and how the TCD settings interrelate.

**Figure 21-7. Memory array terms**

## 21.6.2 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data. Most programming errors are reported on a per channel basis with the exception of channel priority error (ES[CPE]).

For all error types other than group or channel priority errors, the channel number causing the error is recorded in the Error Status register (DMAx\_ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again.

Channel priority errors are identified within a group once that group has been selected as the active group. For example:

1. The eDMA is configured for fixed group and fixed channel arbitration modes.
2. Group 1 is the highest priority and all channels are unique in that group.
3. Group 0 is the next highest priority and has two channels with the same priority level.
4. If Group 1 has any service requests, those requests will be executed.
5. After all of Group 1 requests have completed, Group 0 will be the next active group.
6. If Group 0 has a service request, then an undefined channel in Group 0 will be selected and a channel priority error will occur.
7. This repeats until the all of Group 0 requests have been removed or a higher priority Group 1 request comes in.

In this sequence, for item 2, the eDMA acknowledge lines will assert only if the selected channel is requesting service via the eDMA peripheral request signal. If interrupts are enabled for all channels, the user will get an error interrupt, but the channel number for the ERR register and the error interrupt request line may be wrong because they reflect the selected channel. A group priority error is global and any request in any group will cause a group priority error.

If priority levels are not unique, when any channel requests service, a channel priority error is reported. The highest channel/group priority with an active request is selected, but the lowest numbered channel with that priority is selected by arbitration and executed by the eDMA engine. The hardware service request handshake signals, error interrupts, and error reporting is associated with the selected channel.

### **21.6.3 Arbitration mode considerations**

This section discusses arbitration considerations for the eDMA.

### 21.6.3.1 Fixed group arbitration, Fixed channel arbitration

In this mode, the channel service request from the highest priority channel in the highest priority group is selected to execute. If the eDMA is programmed so that the channels within one group use "fixed" priorities, and that group is assigned the highest "fixed" priority of all groups, that group can take all the bandwidth of the eDMA controller. That is, no other groups will be serviced if there is always at least one DMA request pending on a channel in the highest priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly. Preemption is available in this scenario only.

### 21.6.3.2 Fixed group arbitration, Round-robin channel arbitration

The highest priority group with a request will be serviced. Lower priority groups will be serviced if no pending requests exist in the higher priority groups.

Within each group, channels are serviced starting with the highest channel number and rotating through to the lowest channel number without regard to the channel priority levels assigned within the group.

This scenario could cause the same bandwidth consumption problem as indicated in [Fixed group arbitration, Fixed channel arbitration](#), but all the channels in the highest priority group will be serviced. Service latency will be short on the highest priority group, but could potentially be very much longer as the group priority decreases.

## 21.6.4 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

### 21.6.4.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one ( $TCDn\_CITER = TCDn\_BITER = 1$ ). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the  $TCDn\_CSR[DONE]$  bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are

programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```

TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0

```

This generates the following event sequence:

1. User write to the TCDn\_CSR[START] bit requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCDn\_CSR[DONE] = 0, TCDn\_CSR[START] = 0, TCDn\_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
  - a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
  - b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
  - c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
  - d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
  - e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
  - f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
  - g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
  - h. Write 32-bits to location 0x200C → last iteration of the minor loop → major loop complete.

6. The eDMA engine writes:  $TCDn\_SADDR = 0x1000$ ,  $TCDn\_DADDR = 0x2000$ ,  $TCDn\_CITER = 1$  ( $TCDn\_BITER$ ).
7. The eDMA engine writes:  $TCDn\_CSR[ACTIVE] = 0$ ,  $TCDn\_CSR[DONE] = 1$ ,  $INT[n] = 1$ .
8. The channel retires and the eDMA goes idle or services the next channel.

#### 21.6.4.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled in the ERQ register, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:  $TCDn\_CSR[DONE] = 0$ ,  $TCDn\_CSR[START] = 0$ ,  $TCDn\_CSR[ACTIVE] = 1$ .
4. eDMA engine reads: channel  $TCDn$  data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
  - a. Read byte from location  $0x1000$ , read byte from location  $0x1001$ , read byte from  $0x1002$ , read byte from  $0x1003$ .
  - b. Write 32-bits to location  $0x2000 \rightarrow$  first iteration of the minor loop.
  - c. Read byte from location  $0x1004$ , read byte from location  $0x1005$ , read byte from  $0x1006$ , read byte from  $0x1007$ .
  - d. Write 32-bits to location  $0x2004 \rightarrow$  second iteration of the minor loop.
  - e. Read byte from location  $0x1008$ , read byte from location  $0x1009$ , read byte from  $0x100A$ , read byte from  $0x100B$ .
  - f. Write 32-bits to location  $0x2008 \rightarrow$  third iteration of the minor loop.

- g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
- h. Write 32-bits to location 0x200C → last iteration of the minor loop.
6. eDMA engine writes:  $TCDn\_SADDR = 0x1010$ ,  $TCDn\_DADDR = 0x2010$ ,  $TCDn\_CITER = 1$ .
7. eDMA engine writes:  $TCDn\_CSR[ACTIVE] = 0$ .
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes:  $TCDn\_CSR[DONE] = 0$ ,  $TCDn\_CSR[START] = 0$ ,  $TCDn\_CSR[ACTIVE] = 1$ .
12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:
  - a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
  - b. Write 32-bits to location 0x2010 → first iteration of the minor loop.
  - c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
  - d. Write 32-bits to location 0x2014 → second iteration of the minor loop.
  - e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
  - f. Write 32-bits to location 0x2018 → third iteration of the minor loop.
  - g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
  - h. Write 32-bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes:  $TCDn\_SADDR = 0x1000$ ,  $TCDn\_DADDR = 0x2000$ ,  $TCDn\_CITER = 2$  ( $TCDn\_BITER$ ).
15. eDMA engine writes:  $TCDn\_CSR[ACTIVE] = 0$ ,  $TCDn\_CSR[DONE] = 1$ ,  $INT[n] = 1$ .

16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

### 21.6.4.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567x) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a  $2^4$  byte (16-byte) size queue.

**Table 21-5. Modulo example**

Transfer Number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

### 21.6.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

#### 21.6.5.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the `TCDn_CITER` field and test for a change. Another method may be extracted from the sequence shown below. The second method is to test the `TCDn_CSR[START]` bit and the `TCDn_CSR[ACTIVE]` bit. The minor-loop-

complete condition is indicated by both bits reading zero after the TCD $n$ \_CSR[START] was set. Polling the TCD $n$ \_CSR[ACTIVE] bit may be inconclusive, because the active status may be missed if the channel execution is short in duration.

The TCD status bits execute the following sequence for a software activated channel:

Stage	TCD $n$ _CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD $n$ \_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD $n$ _CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the TCD $n$ \_CSR[DONE] bit.

The TCD $n$ \_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

## 21.6.5.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCD $n$ \_SADDR, TCD $n$ \_DADDR, and TCD $n$ \_NBYTES values if read while a channel executes. The true values of the SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file and not the values in the TCD local memory for that channel. The addresses, SADDR and

DADDR, and NBYTES, which decrement to zero as the transfer progresses, can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

### 21.6.5.3 Checking channel preemption status

Preemption is available only when fixed arbitration is selected for both group and channel arbitration modes. A preemptive situation is one in which a preempt-enabled channel runs and a higher priority request becomes active. When the eDMA engine is not operating in fixed group, fixed channel arbitration mode, the determination of the actively running relative priority outstanding requests become undefined. Channel and/or group priorities are treated as equal, that is, constantly rotating, when Round-Robin Arbitration mode is selected.

The TCD $n$ \_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two TCD $n$ \_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

### 21.6.6 Channel Linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD $n$ \_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the EDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD $n$ \_CITER[E\_LINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER [E_LINK] = 1
TCDn_CITER [LINKCH] = 0xC
TCDn_CITER [CITER] value = 0x4
TCDn_CSR [MAJOR_E_LINK] = 1
TCDn_CSR [MAJOR_LINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12\_CSR[START] bit

2. Minor loop done → set TCD12\_CSR[START] bit
3. Minor loop done → set TCD12\_CSR[START] bit
4. Minor loop done, major loop done → set TCD7\_CSR[START] bit

When minor loop linking is enabled ( $TCDn\_CITER[E\_LINK] = 1$ ), the  $TCDn\_CITER[CITER]$  field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled ( $TCDn\_CITER[E\_LINK] = 0$ ), the  $TCDn\_CITER[CITER]$  field uses a 15-bit vector to form the current iteration count. The bits associated with the  $TCDn\_CITER[LINKCH]$  field are concatenated onto the CITER value to increase the range of the CITER.

### Note

The  $TCDn\_CITER[E\_LINK]$  bit and the  $TCDn\_BITER[E\_LINK]$  bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

**Table 21-6. Channel Linking Parameters**

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	CITER[E_LINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	CSR[MAJOR_E_LINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

## 21.6.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

### 21.6.7.1 Dynamically changing the channel priority

The following two options are recommended for dynamically changing channel priority levels:

1. Switch to Round-Robin Channel Arbitration mode, change the channel priorities, then switch back to Fixed Arbitration mode,
2. Disable all the channels, change the channel priorities, then enable the appropriate channels.

### 21.6.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCD.major.e\_link bit during channel execution (see the diagram in [TCD structure](#)). This bit is read from the TCD local memory at the end of channel execution, thus allowing the user to enable the feature during channel execution.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCD.major.e\_link bit at the same time the eDMA engine is retiring the channel. The TCD.major.e\_link would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCD.major.e\_link bit.
2. Read back the TCD.major.e\_link bit.
3. Test the TCD.major.e\_link request status:
  - If TCD.major.e\_link = 1, the dynamic link attempt was successful.
  - If TCD.major.e\_link = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCD.major.e\_link bit to zero on any writes to a channel's TCD.word7 after that channel's TCD.done bit is set, indicating the major loop is complete.

#### NOTE

The user must clear the TCD.done bit before writing the TCD.major.e\_link bit. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

### 21.6.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in eDMA programmer's model, thus replacing the current descriptor.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic scatter/gather operation by enabling the TCD.e\_sg bit at the same time the eDMA engine is retiring the channel. The TCD.e\_sg would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods for this coherency model are shown in the following subsections. Method 1 has the advantage of reading the major.linkch field and the e\_sg bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD.major.e\_link and TCD.e\_sg bits to zero on any writes to a channel's TCD.word7 if that channel's TCD.done bit is set indicating the major loop is complete.

#### NOTE

The user must clear the TCD.done bit before writing the TCD.major.e\_link or TCD.e\_sg bits. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

#### 21.6.7.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCD.major.e\_link bit is zero, the TCD.major.linkch field is not used by the eDMA. In this case, the TCD.major.linkch bits may be used for other purposes. This method uses the TCD.major.linkch field as a TCD identification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCD.major.linkch field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to the TCD.d\_req bit.

Should a dynamic scatter/gather attempt fail, setting the TCD.d\_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offset value.

3. Write the TCD.dlast\_sga field with the scatter/gather address.
4. Write 1b to the TCD.e\_sg bit.
5. Read back the 16 bit TCD control/status field.
6. Test the TCD.e\_sg request status and TCD.major.linkch value:

If e\_sg = 1b, the dynamic link attempt was successful.

If e\_sg = 0b and the major.linkch (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

If e\_sg = 0b and the major.linkch (ID) changed, the dynamic link attempt was successful (the new TCD's e\_sg value cleared the e\_sg bit).

### 21.6.7.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCD.dlast\_sga field as a TCD identification (ID).

1. Write 1b to the TCD.d\_req bit.

Should a dynamic scatter/gather attempt fail, setting the d\_req bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offset value.

2. Write the TCD.dlast\_sga field with the scatter/gather address.
3. Write 1b to the TCD.e\_sg bit.
4. Read back the TCD.e\_sg bit.
5. Test the TCD.e\_sg request status:

If e\_sg = 1b, the dynamic link attempt was successful.

If e\_sg = 0b, read the 32 bit TCD dlast\_sga field.

If e\_sg = 0b and the dlast\_sga did not change, the attempted dynamic link did not succeed (the channel was already retiring).

If e\_sg = 0b and the dlast\_sga changed, the dynamic link attempt was successful (the new TCD's e\_sg value cleared the e\_sg bit).

## 21.6.8 Suspend/resume a DMA channel with active hardware service requests

The DMA allows the user to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. Once the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (DSPI), ADC, or other module.

### 21.6.8.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check [Hardware Request Status Register \(DMA\\_HRS\)](#) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on the appropriate DMA channel.

### 21.6.8.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ bit.
2. Enable the DMA service request at the peripheral.

For example, assume a DSPI module is set as a master for transmitting data via a DMA service request when the DSPI\_TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If the user needs to suspend the DMA/DSPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI\_RSER[TFFF\_RE] . Confirm that DSPI\_RSER[TFFF\_RE] is 0.

2. Ensure there is no DMA service request from the DSPI by verifying that DMA\_HRS[HRS $n$ ] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.



# **Chapter 22**

## **Enhanced Secured Digital Host Controller**

### **22.1 Overview**

The enhanced secured digital host controller (eSDHC) provides interface between the host system and the SD/SDIO/MMC cards, as depicted in the figure below.

The eSDHC acts as a bridge, passing host bus transactions to the SD/SDIO/MMC cards by sending commands and performing data accesses to/from the cards.

It handles the SD/SDIO/MMC protocols at the transmission level.

Different types of cards supported by the eSDHC are described below:

- MultiMediaCard (MMC)

MMC is a universal low-cost data storage and communication medium designed to cover a wide area of applications, including mobile video and gaming, which are available from either pre-loaded MMC cards or downloadable from cellular phones, WLAN, or other wireless networks. Old MMC cards are based on a seven-pin serial bus with a single data pin, while the new high-speed MMC communication is based on advanced 11-pin serial bus designed to operate in a low voltage range.

- Secure digital (SD) card

The secure digital (SD) card is an evolution over the old MMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in the emerging audio and video consumer electronic devices. The physical form factor, pin assignments, and data transfer protocol are forward-compatible with the old MMC. The chip supports SDXC card.

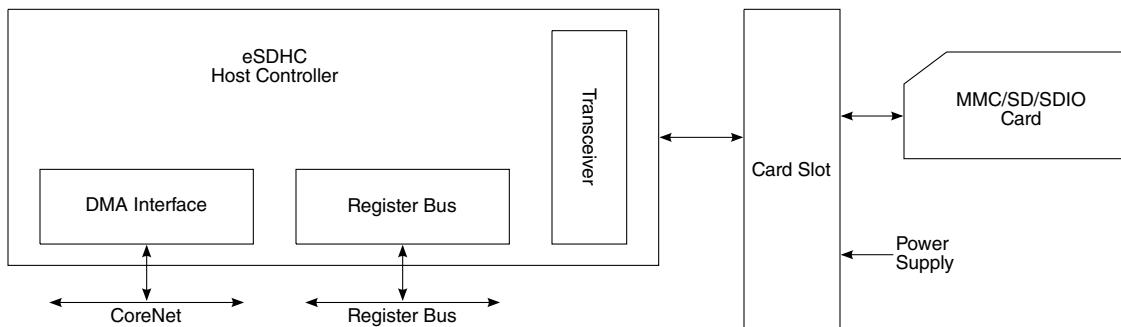
- SDIO

Under the SD protocol, the SD cards can be categorized as a memory card, I/O card, or combo card. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard. The I/O card provides high-speed

data I/O with low power consumption for mobile electronic devices. The combo card has both memory and I/O functions. For the sake of simplicity, the following figure does not show cards with reduced sizes.

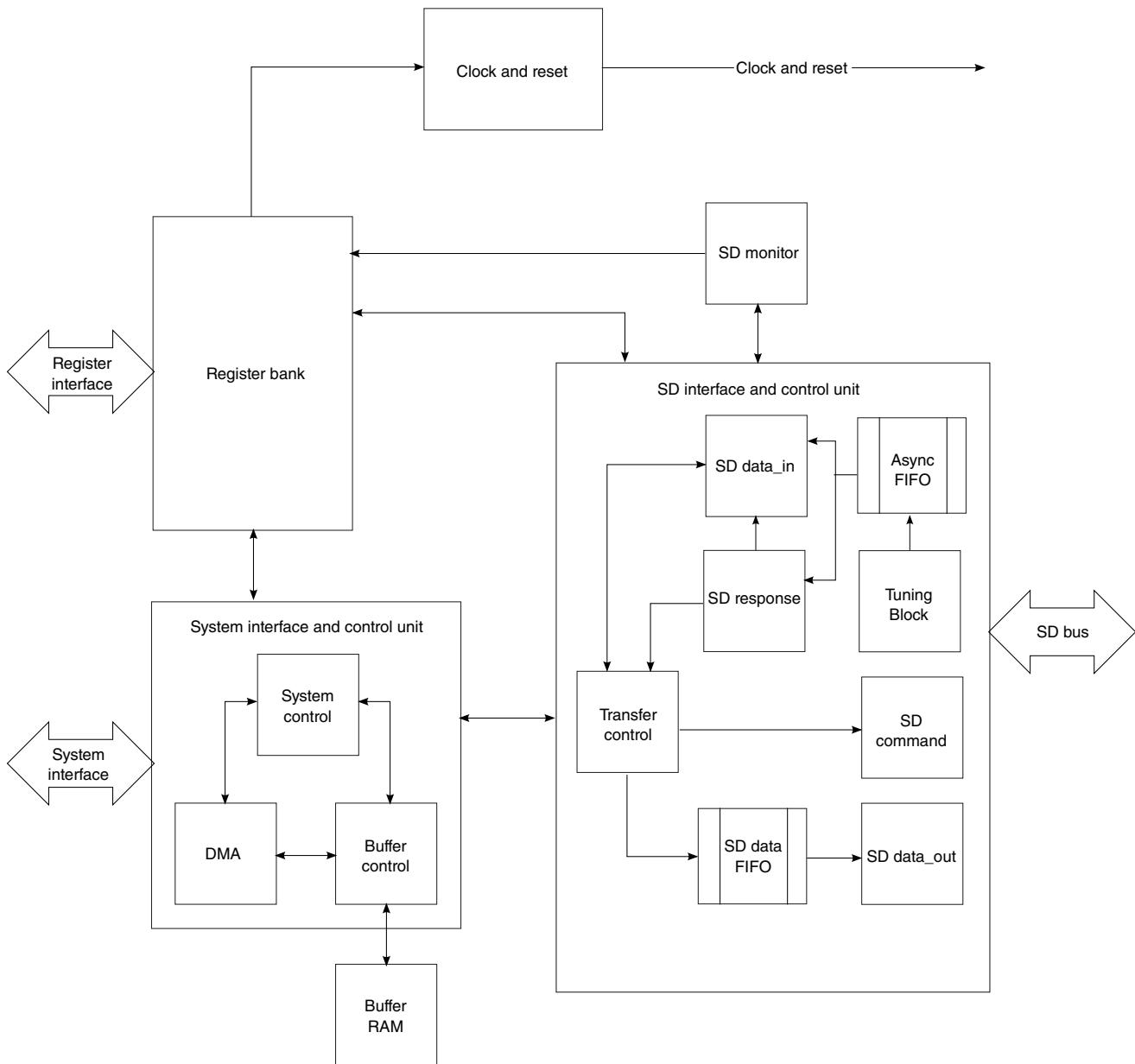
In addition, the chip also supports the embedded devices, such as and eSDIO. Note that all references to SDIO are also applicable to eSDIO.

The eSDHC acts as a bridge, passing host bus transactions to MMC/SD/SDIO cards by sending commands and performing data accesses to or from the cards. It handles the MMC/SD/SDIO protocol at the transmission level. The figure below shows connection of the eSDHC.



**Figure 22-1. System connection of the eSDHC**

The figure below is a high-level block diagram of eSDHC.



**Figure 22-2. Enhanced secure digital host controller block diagram**

### 22.1.1 eSDHC features summary

The features of the eSDHC module include the following:

- Conforms to the SD Host controller standard specification version 3.0, including test event register support
- Compatible with the eMMC system specification version 4.5
- Compatible with the SD memory card specification version 3.01, and supports the high capacity SD memory card

- Compatible with the SDIO card specification version 3.0
- Designed to work with SD memory, SDIO, SD combo, MMC, and their variants, such as mini and micro.
- MMCHC (MMC high capacity) support
- Supports 1- or 4-bit SD and SDIO modes, 1- or 4- or 8-bit MMC modes
- Supports single-block and multi-block read, and write data transfer
- Supports block sizes of 1-2048 bytes
- Supports the mechanical write protect detection. In the case where write protect is enabled, the host will not initiate any write data command to the card
- Supports card detection from SDHC\_CD\_B
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO read wait and suspend resume operations
- Supports Auto CMD12 and Auto CMD23 for multi-block transfer
- Host can initiate command that does not use data lines, while data transfer is in progress
- Allows card to interrupt the host in 1 bit and 4 bit SDIO modes, also supports interrupt period
- Embodies a configurable 128 x 32 bit FIFO for read/write data
- Supports SDMA, ADMA1, and ADMA2 capabilities
- Host will send 80 idle SD clock cycles to card, which are needed during card power-up, if bit INITA in the system control register (SYSCTL) is set
- Supports SDIO asynchronous interrupt
- Supports clock divider with finer granularity, that is, with values 1,2,3....1024 or 1,2,4,8...2048
- Supports standard, high and extended capacity card types
- Supports SD UHS-1 speed modes: SDR12, SDR25, SDR50, SDR104, DDR50
- Supports MMC high speed, HS200 and DDR mode

## **22.1.2 Modes and operations**

### **22.1.2.1 Data transfer modes**

The eSDHC can select the following modes for data transfer:

- SD 1 bit
- SD 4 bit
- MMC 1 bit
- MMC 4 bit
- MMC 8 bit

- Identification mode
- MMC full speed mode
- MMC high speed mode
- SD/SDIO full speed mode
- SD/SDIO high speed mode
- SD/SDIO UHS-1 SDR12 mode
- SD/SDIO UHS-1 SDR25 mode
- SD/SDIO UHS-1 SDR50 mode
- SD/SDIO UHS-1 SDR104 mode
- SD/SDIO UHS-1 DDR50 mode
- MMC HS200 mode
- MMC DDR mode

## **NOTE**

For the maximum supported speed of the modes, refer chip-specific datasheet.

## **22.2 External signals**

### **22.2.1 External signals overview**

The eSDHC has the following associated I/O signals:

- SDHC\_CLK is an internally generated clock signal that drives the SD/SDIO/MMC card.
- SDHC\_CLK\_SYNC\_OUT has same frequency as SDHC\_CLK. It should be sent out on board for loop back from close to external card.
- SDHC\_CLK\_SYNC\_IN has same frequency as SDHC\_CLK. It is loopback input from close to card on board.
- SDHC\_CMD I/O sends commands to and receives responses from the card.
- SDHC\_DAT[7:0] performs data transfers between the eSDHC and the card. If the eSDHC is desired to support a 4 bit data transfer, DAT[7:4] can also be optional and tied high.
- SDHC\_CD\_B and SDHC\_WP are card detection and write protection signals from the socket.
- SDHC\_CMD\_DIR is an output signal used to control the direction of CMD line in an external voltage translator.
- SDHC\_DAT0\_DIR is an output signal used to control the direction of DAT0 line in an external voltage translator.

## External signals

- SDHC\_DAT123\_DIR is an output signal used to control the direction of DAT1-DAT3 lines in an external voltage translator.
- SDHC\_VS is an output signal used to control the voltage of external SD bus supply.

### 22.2.2 eSDHC signal descriptions

The following table describes eSDHC signals.

**Table 22-1. Signal properties**

Name	Port	Function	Reset state	Pull up
SDHC_CLK	O	Clock for SD/SDIO/MMC card or eMMC device	0	N/A
SDHC_CLK_SYNC_OUT	O	Clock to loopback from close to card	0	N/A
SDHC_CLK_SYNC_IN	I	Clock looped back from card	0	N/A
SDHC_CMD	I/O	CMD line connect to card	1	Pull up
SDHC_CMD_DIR	O	CMD line direction control	0	N/A
SDHC_DAT7	I/O	DAT7 line in 8-bit mode Not used in other modes	1	Pull up
SDHC_DAT6	I/O	DAT6 line in 8-bit mode Not used in other modes	1	Pull up
SDHC_DAT5	I/O	DAT5 line in 8-bit mode Not used in other modes	1	Pull up
SDHC_DAT4	I/O	DAT4 line in 8-bit mode Not used in other modes	1	Pull up
SDHC_DAT3	I/O	DAT3 line in 4/8-bit mode	0	Pull up. Do not use DAT3 pin as a CD pin.
SDHC_DAT2	I/O	DAT2 line or read wait in 4/8-bit mode Read wait in 1-bit mode	1	Pull up
SDHC_DAT1	I/O	DAT1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	1	Pull up
SDHC_DAT0	I/O	DAT0 line in all modes Also used to detect busy state	1	Pull up
SDHC_DAT0_DIR	O	DAT0 line direction control	0	N/A
SDHC_CD_B	I	Card detection pin If not used tie high	N/A	N/A
SDHC_WP	I	Card write protect detect If not used tie low	N/A	N/A
SDHC_VS	O	Control the voltage on SD pads to be high voltage (around 3.0V) or low voltage (around 1.8V). '0' stands for high voltage range	0	N/A

*Table continues on the next page...*

**Table 22-1. Signal properties (continued)**

Name	Port	Function	Reset state	Pull up
		Optional output		
SDHC_DAT123_DIR	O	DAT1-DAT3 lines direction control	0	N/A

## 22.3 eSDHC register descriptions

The table below shows the memory-mapped registers of the eSDHC module and lists the offset, name, and a cross-reference to the complete description of each register. These register only support 32-bit accesses.

### 22.3.1 eSDHC memory map

eSDHC base address: 156\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SDMA system address register/Block attributes 2 (DSADDR_BLKAT TR2)	32	RW	0000_0000h
4h	Block attributes register (BLKATTR)	32	RW	0000_0000h
8h	Command argument register (CMDARG)	32	RW	0000_0000h
Ch	Transfer type register (XFERTYP)	32	RW	0000_0000h
10h	Command response 0 register (CMDRSP0)	32	RO	0000_0000h
14h	Command response 1 register (CMDRSP1)	32	RO	0000_0000h
18h	Command response 2 register (CMDRSP2)	32	RO	0000_0000h
1Ch	Command Response 3 register (CMDRSP3)	32	RO	0000_0000h
20h	Buffer data port register (DATPORT)	32	RW	0000_0000h
24h	Present state register (PRSSTAT)	32	RO	<a href="#">Table 22-4</a>
28h	Protocol control register (PROCTL)	32	RW	0000_0020h
2Ch	System Control Register when ESDHCCTL[CRS=0] (SYSCTL_E SDHCCTL_CRS_0)	32	RW	0000_8038h
2Ch	System Control Register when ESDHCCTL[CRS=1] (SYSCTL_E SDHCCTL_CRS_1)	32	RW	0000_0000h
30h	Interrupt status register (IRQSTAT)	32	W1C	0000_0000h
34h	Interrupt status enable register (IRQSTATEN)	32	RW	377F_11FFh
38h	Interrupt signal enable register (IRQSIGEN)	32	RW	0400_1000h

Table continues on the next page...

## eSDHC register descriptions

Offset	Register	Width (In bits)	Access	Reset value
3Ch	Auto CMD Error Status Register / System Control 2 Register (AUTO_CERR_SYSCTL2)	32	RW	0000_0000h
40h	Host controller capabilities register (HOSTCAPBLT)	32	RO	35F2_0000h
44h	Watermark level register (WML)	32	RW	0010_0010h
50h	Force event register (FEVT)	32	WO	0000_0000h
54h	ADMA error status register (ADMAES)	32	RO	0000_0000h
58h	ADMA system address register (ADSADDR)	32	RW	0000_0000h
FCh	Host controller version register (HOSTVER)	32	RO	0000_2102h
104h	DMA error address register (DMAERRADDR)	32	RO	0000_0000h
10Ch	DMA error attribute register (DMAERRATTR)	32	RO	0000_0000h
114h	Host controller capabilities register 2 (HOSTCAPBLT2)	32	RO	0000_AF07h
120h	Tuning block control register (TBCTL)	32	RW	0000_0002h
124h	Tuning block status register (TBSTAT)	32	RW	0000_0000h
128h	Tuning block pointer register (TBPTR)	32	RW	0000_0000h
140h	SD direction control register (SDDIRCTL)	32	RW	0000_0001h
144h	SD Clock Control Register (SDCLKCTL)	32	RW	0000_0000h
40Ch	eSDHC control register (ESDHCCCTL)	32	RW	0000_0000h

## 22.3.2 SDMA system address register/Block attributes 2 (DSADR\_BLKATTR2)

### 22.3.2.1 Offset

Register	Offset
DSADDR_BLKATTR2	0h

### 22.3.2.2 Function

The DSADDR contains the physical system memory address used for single DMA transfers or second argument for Auto CMD23.

### 22.3.2.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									DS_ADDR							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									DS_ADDR							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.2.4 Fields

Field	Function
0-31 DS_ADDR	<p>DMA system address / Block attributes 2</p> <p>This register contains the physical system memory address used for DMA transfers or second argument for Auto CMD23.</p> <p>SDMA system address:</p> <p>This register contains the 32-bit system memory address for a single DMA (SDMA) transfer. When the eSDHC stops a DMA transfer, this register points to the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). The host driver should initialize this register before starting a every DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register.</p> <p>The eSDHC DMA does not support a virtual memory system. It only supports continuous physical memory access.</p> <p>Block attributes 2:</p> <p>This register is used with Auto CMD23 to set 32-bit block count value to the argument of CMD23 while executing Auto CMD23.</p> <p>If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used.</p> <p>If Auto CMD23 is used without ADMA, the available block count value is limited by 16-bit Block Count field in Block Attributes register. 65536 blocks is the maximum value available in this case.</p>

### 22.3.3 Block attributes register (BLKATTR)

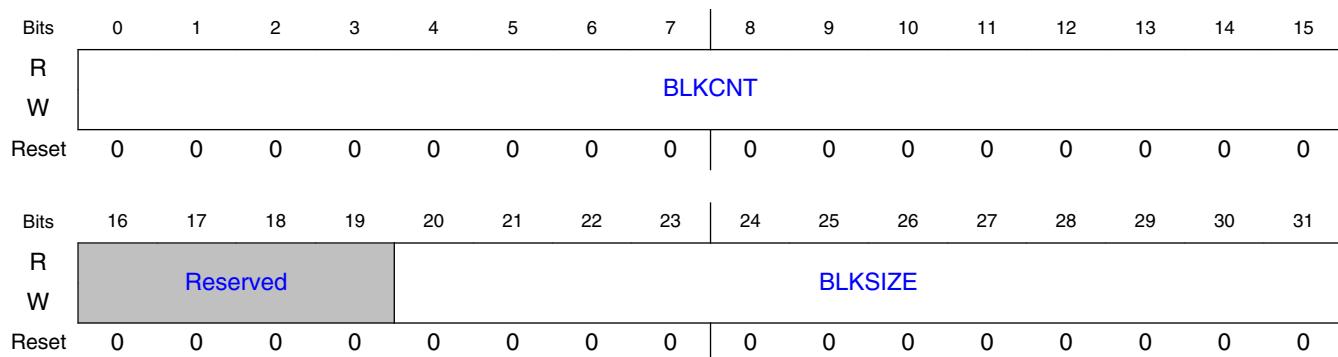
### 22.3.3.1 Offset

Register	Offset
BLKATTR	4h

### 22.3.3.2 Function

The BLKATTR is used to configure the number of data blocks and the number of bytes in each block.

### 22.3.3.3 Diagram



### 22.3.3.4 Fields

Field	Function
0-15 BLKCNT	<p>Blocks count for current transfer.</p> <ul style="list-style-type: none"> <li>This register is enabled when XFERTYP[BCEN] is set to 1 and is valid only for multiple block transfers. The host driver should set this register to a value between 1 and the maximum block count. The eSDHC decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to 0 results in no data blocks being transferred.</li> <li>This register should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored.</li> <li>When saving transfer content as a result of a suspend command, the number of blocks yet to be transferred can be determined by reading this register. The reading of this register should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when suspend command is sent out, eSDHC will regard the current transfer is aborted and change BLKCNT back to its original value instead of keeping the dynamical indicator of remained block count.</li> <li>When restoring transfer content prior to issuing a resume command, the host driver should restore the previously saved block count.</li> </ul>

Table continues on the next page...

Field	Function
	0000000000000000b - Stop count 0000000000000001b - 1 block 0000000000000010b - 2 blocks 111111111111111b - 65535 blocks
16-19 —	Reserved
20-31 BLKSIZE	Transfer block size. This register specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations will be ignored. 00000000000b - No data transfer 000000000001b - 1 byte 0000000000010b - 2 bytes 0000000000011b - 3 bytes 00000000000100b - 4 bytes 00111111111b - 511 bytes 001000000000b - 512 bytes 100000000000b - 2048 bytes

## 22.3.4 Command argument register (CMDARG)

### 22.3.4.1 Offset

Register	Offset
CMDARG	8h

### 22.3.4.2 Function

The CMDARG contains the SD/MMC command argument.

### 22.3.4.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									CMDARG							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CMDARG							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.4.4 Fields

Field	Function
0-31 CMDARG	Command argument. The SD/MMC command argument is specified as bits 39-8 of the command format in the SD or MMC Specification. This register is write protected when PRSSTAT[CIHB] is set.

## 22.3.5 Transfer type register (XFERTYP)

### 22.3.5.1 Offset

Register	Offset
XFERTYP	Ch

### 22.3.5.2 Function

The host driver should set the XFERTYP to issue any new command. To prevent data loss, the eSDHC prevents writing to the bits, that are involved in the data transfer of this register, when data transfer is active: MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver should check PRSSTAT[CDIHB] and PRSSTAT[CIHB] before writing to this register. When PRSSTAT[CDIHB] is set, any attempt to send a command with data by writing to this register is ignored; when PRSSTAT[CIHB] is set, any write to this register is ignored.

On sending commands with data transfer, it is mandatory that the block size is non-zero. Besides, block count must also be non-zero, or indicated as single block transfer (XFERTYP[MSBSEL] is '0' when written), or block count is disabled (XFERTYP[BCEN] is '0' when written), otherwise eSDHC will ignore the sending of this command and do nothing. For write commands, with all above restrictions, it is also mandatory that the write protect switch is not active (PRSSTAT[WPS] is '1'), otherwise eSDHC will also ignore the command.

If the commands with write data transfer does not receive the response in 64 clock cycles, that is, response time-out, eSDHC will regard the external device does not accept the command and will not initiate the data transfer on SD bus. In this scenario, the driver should perform error recovery and issue the command again to re-try the transfer.

It is also possible that for some reason the card responds to the read data command but eSDHC does not receive the response, and if it is a DMA (SDMA or ADMA ) read operation, the external system memory is over-written by the DMA with data sent back from the card.

**Table 22-2. Transfer Type Register Setting for Various Transfer Types**

Multi-/Single Block Select	Block Count Enable	Block Count	Function
0	Don't Care	Don't Care	Single Transfer
1	0	Don't Care	Infinite Transfer
1	1	Positive Number	Multiple Transfer
1	1	Zero	No Data Transfer

The table below shows the relationship between the command index check enable and the command CRC check enable, in regards to the response type bits as well as the name of the response type.

**Table 22-3. Relationship Between Parameters and the Name of the Response Type**

Response type	Index Check Enable	CRC Check Enable	Name of Response Type
00	0	0	No Response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6, R7
11	1	1	R1b, R5b

## eSDHC register descriptions

- In the SDIO Specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO Specification. But R5b is defined in this specification to specify that the eSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command should be used with R5b.
- The CRC field for R3 and R4 is expected to be all 1 bits. The CRC check should be disabled for these response types.

### 22.3.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			CMDINX					CMDTYP							
W										DPSE_L		CICEN		CCCEN		RSPTY_P
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved									MSBSE_L		DTDSE_L		ACEN		DMAEN
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.5.4 Fields

Field	Function
0-1	Reserved
—	
2-7 CMDINX	Command index Command index. These bits should be set to the command number that is specified in bits 45-40 of the command format in the SD Memory Card Physical Layer Specification and SDIO Card Specification .
8-9 CMDTYP	Command Type Command Type. There are three types of special commands: suspend, resume, and abort. These bits should be set to 00b for all other commands. <ul style="list-style-type: none"> <li>Suspend command: If the suspend command succeeds, the eSDHC should assume that the card bus has been released and that it is possible to issue the next command which uses the DAT line. Since eSDHC does not monitor the content of command response, it does not know if the suspend command succeeded or not. It is the host driver's responsibility to check the status of the suspend command and send another command marked as Suspend to inform the eSDHC that a suspend command was successfully issued. Refer to <a href="#">Suspend resume</a> for more details. After the start bit of command is sent, the eSDHC de-asserts read wait for read transactions and stops checking busy for write transactions. In 4-bit mode, the interrupt cycle starts. If the suspend command fails, the</li> </ul>

Table continues on the next page...

Field	Function
	<p>eSDHC will maintain its current state, and the host driver should restart the transfer by setting PROCTL[CREQ].</p> <ul style="list-style-type: none"> <li>Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the suspend command and then sends the resume command.</li> <li>Abort command: If this command is set when executing a read transfer, the eSDHC will stop write to the buffer after end bit of abort command is sent. If this command is set when executing a write transfer, the eSDHC will stop driving the DAT line. After issuing the abort command, the host driver should issue a software reset (abort transaction).</li> </ul> <p>00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O abort in CCCR</p>
10 DPSEL	<p>Data present select. This bit is set to 1 to indicate that data is present and should be transferred using the DAT line. It is set to 0 for the following:</p> <ul style="list-style-type: none"> <li>Commands using only the CMD line (for example, CMD52)</li> <li>Commands with no data transfer, but using the busy signal on DAT[0] line (R1b or R5b, for example, CMD38)</li> </ul> <p><b>NOTE:</b> In resume command, this bit should be set, and other bits in this register should be set the same as when the transfer was initially launched. When the write protect switch is on (that is, PRSSTAT[WPS] is active as '0'), any command with a write operation will be ignored. That is to say, when this bit is set, while XFRTYP[DTDSEL] is 0, writes to the transfer type register (XFRTYP) are ignored.</p> <p>0b - No data present 1b - Data present</p>
11 CICEN	<p>Command index check enable. If this bit is set to 1, the eSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a command index error. If this bit is set to 0, the Index field is not checked.</p> <p>0b - Disable 1b - Enable</p>
12 CCCEN	<p>Command CRC check enable</p> <p>Command CRC check enable. If this bit is set to 1, the eSDHC should check the CRC field in the response. If an error is detected, it is reported as a command CRC error. If this bit is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. (Refer to RSPTYP[1:0] and <a href="#">Table 22-3</a>.)</p> <p>0b - Disable 1b - Enable</p>
13 —	Reserved
14-15 RSPTYP	<p>Response type select.</p> <p>00b - No response 01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response</p>
16-25 —	Reserved
26 MSBSEL	<p>Multi-/single-block select</p> <p>Multi-/single-block select. This bit enables multiple block DAT line data transfers. For any other commands, this bit should be set to 0. If this bit is 0, it is not necessary to set the block count register. (Refer to <a href="#">Table 22-2</a>.)</p>

Table continues on the next page...

## eSDHC register descriptions

Field	Function
	0b - Single block 1b - Multiple blocks
27 DTDSEL	Data transfer direction select. This bit defines the direction of DAT line data transfers. The bit is set to 1 by the host driver to transfer data from the SD card to the eSDHC and is set to 0 for all other commands.  0b - Write (host to card) 1b - Read (card to host)
28-29 ACEN	Auto CMD12 enable  Auto CMD12 enable. Multiple block transfers for memory require a CMD12 to stop the transaction. When this bit is set to 1, the eSDHC will issue a CMD12 automatically when the last block transfer has completed. The host driver should not set this bit to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File security specification do not require CMD12. In single block transfer, the eSDHC will ignore this bit whether it is set or not.  Auto CMD23 Enable. When this bit is set to 10b, the eSDHC will issue a CMD23 automatically before issuing a command specified in the transfer type register. The following conditions are required to use Auto CMD23: <ul style="list-style-type: none"><li>• A memory card that supports CMD23 (For SD card, SCR[33]=1)</li><li>• If DMA is used, it shall be ADMA</li><li>• Only when CMD18 or CMD25 is issued</li></ul> <b>NOTE:</b> eSDHC does not check command index  Auto CMD23 can be issued with or without ADMA, by writing Transfer Type register, eSDHC issues a CMD23 first and then issues a command specified by Transfer Type register. If response errors of CMD23 are detected, the second command is not issued. A CMD23 error is indicated in the Auto CMD Error Status register.  32-bit block count value for CMD23 is set to DMA System Address / Argument 2 register.  00b - Auto CMD Disable 01b - Auto CMD12 Enable 10b - Auto CMD23 Enable 11b - Reserved
30 BCEN	Block count enable. This bit is used to enable the block count register, which is only relevant for multiple block transfers. When this bit is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer.  If ADMA data transfer is more than 65535 blocks, this bit shall be set to 0. In this case, data transfer length is designated by Descriptor Table.  0b - Disable 1b - Enable
31 DMAEN	DMA enable  DMA enable. This bit enables DMA functionality. If this bit is set to 1, a DMA operation should begin when the host driver sets the DPSEL bit of this register. Whether the single DMA or ADMA is active depends on PROCTL[DMAS].  0b - Disable 1b - Enable

## 22.3.6 Command response 0 register (CMDRSP0)

### 22.3.6.1 Offset

Register	Offset
CMDRSP0	10h

### 22.3.6.2 Function

The CMDRSP0 is used to store part 0 of the response bits from the card.

### 22.3.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.6.4 Fields

Field	Function
0-31	Command response 0
CMDRSP0	Command response 0. Refer to <a href="#">Command Response 3 register (CMDRSP3)</a> for the mapping of command responses from the SD bus to this register for each response type.

### 22.3.7 Command response 1 register (CMDRSP1)

### 22.3.7.1 Offset

Register	Offset
CMDRSP1	14h

### 22.3.7.2 Function

The CMDRSP1 is used to store part 1 of the response bits from the card.

### 22.3.7.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.7.4 Fields

Field	Function
0-31	Command response 1
CMDRSP1	Command response 1. Refer to <a href="#">Command Response 3 register (CMDRSP3)</a> for the mapping of command responses from the SD bus to this register for each response type.

## 22.3.8 Command response 2 register (CMDRSP2)

### 22.3.8.1 Offset

Register	Offset
CMDRSP2	18h

### 22.3.8.2 Function

The CMDRSP2 is used to store part 2 of the response bits from the card.

### 22.3.8.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.8.4 Fields

Field	Function
0-31	Command response 2
CMDRSP2	Command response 2. Refer to <a href="#">Command Response 3 register (CMDRSP3)</a> for the mapping of command responses from the SD bus to this register for each response type.

## 22.3.9 Command Response 3 register (CMDRSP3)

### 22.3.9.1 Offset

Register	Offset
CMDRSP3	1Ch

### 22.3.9.2 Function

The CMDRSP3 is used to store part 3 of the response bits from the card.

Table below describes the mapping of command responses from the SD bus to command response registers for each response type. In the table, R *n* refers to a bit range within the response data as transmitted on the SD bus.

**Table 22-4. Response bit definition for each response type**

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card Status	R[39:8]	CMDRSP0
R1b (Auto CMD12 response)	Card Status for Auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[8:31], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (Publish RCA)	New Published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows the following:

Most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, the eSDHC only stores part of the response data in the command response registers (CMDRSP *n*). This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the index field and the CRC, are checked by the eSDHC (as specified by the command index check enable and the command CRC check enable

bits in the transfer type register, XFERTYP) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, the eSDHC checks R[47:1], and if the response length is 136 the eSDHC checks R[119:1].

Since eSDHC may have a multiple block data transfer executing concurrently with a CMD\_wo\_DAT command, it stores the Auto CMD12 response in the CMDRSP3 register. The CMD\_wo\_DAT response is stored in CMDRSP0. This allows the eSDHC to avoid overwriting the Auto CMD12 response with the CMD\_wo\_DAT and vice versa. When the eSDHC modifies part of the command response registers (CMDRSP $n$ ), as shown in the table above, it preserves the unmodified bits.

### 22.3.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.9.4 Fields

Field	Function
0-31	Command response 3
CMDRSP3	Command response 3. Refer to <a href="#">Command Response 3 register (CMDRSP3)</a> for the mapping of command responses from the SD bus to this register for each response type.

### 22.3.10 Buffer data port register (DATPORT)

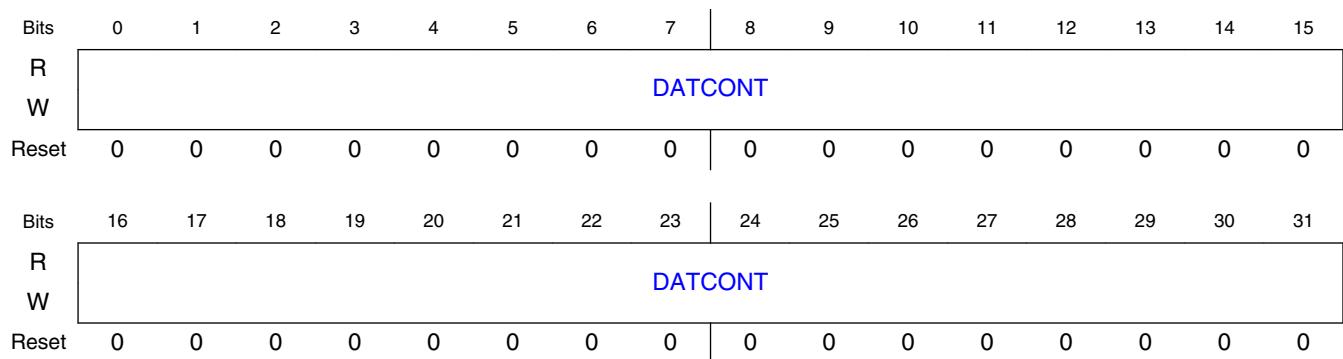
### 22.3.10.1 Offset

Register	Offset
DATPORT	20h

### 22.3.10.2 Function

The DATPORT is a 32-bit data port register used to access the internal buffer. Byte access is not allowed.

### 22.3.10.3 Diagram



### 22.3.10.4 Fields

Field	Function
0-31 DATCONT	Data content. The buffer data port register is for 32-bit data access by the CPU. When the DMA is enabled, any write to this register is ignored, and any read from this register always yields zeros.

## 22.3.11 Present state register (PRSSTAT)

### 22.3.11.1 Offset

Register	Offset
PRSSTAT	24h

### 22.3.11.2 Function

The host driver can get the status of the eSDHC from the PRSSTAT, which is a 32-bit, read-only register.

### 22.3.11.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DLS L								CLS L		Reserved		VVP S	CDS	Reserved	CINS
W																
Reset	1	1	1	1	1	1	1	1	1	0	0	0	u	u	0	u

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				BRE N	BWEN	RTA	WTA	SDOF F	Reserved			SDST B	DLA	CDIB	CIIB
W	Reserved	0	0	0	0	0	0	0	u	0	0	0	1	0	0	0
Reset	0	0	0	0	0	0	0	0	u	0	0	0	0	0	0	0

### 22.3.11.4 Fields

Field	Function
0-7 DLSL	DAT[7:0] line signal level  DAT[7:0] line signal level. This status is used to check the DAT line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DAT[0]. The reset value is effected by the external pull-up/pull-down resistors. By default, the read value of this bit field after reset is 8'b11111111  DLSL[0]: Data 0 line signal level DLSL[1]: Data 1 line signal level DLSL[2]: Data 2 line signal level DLSL[3]: Data 3 line signal level DLSL[4]: Data 4 line signal level

Table continues on the next page...

## eSDHC register descriptions

Field	Function
	DLSL[5]: Data 5 line signal level DLSL[6]: Data 6 line signal level DLSL[7]: Data 7 line signal level
8 CLSL	CMD line signal level. This status is used to check the CMD line level to recover from errors, and for debugging. The reset value is effected by the external pull-up/pull-down resistor, by default, the read value of this bit after reset is 1, when the command line is pulled up.
9-11 —	Reserved
12 WPS	Write protect state  Write protect state. The write protect switch is supported for memory and combo cards. This bit reflects the write protect state depending on value of SDHC_WP pin of the card socket. A software reset does not affect this bit. The reset value is effected by the external write protect switch. If the SDHC_WP pin is not used, it should be tied low, so that the reset value of this bit is high and write is enabled.  0b - Write protected (SDHC_WP =1) 1b - Write enabled (SDHC_WP =0)
13 CDS	Card detect state  Card detect state. This bit reflects the card inserted/removed state depending on value of the SDHC_CD_B pin for the card socket. Debouncing is not performed on this bit. This bit may be valid, but is not guaranteed, because of propagation delay. Use of this bit is limited to testing since it must be debounced by software. A software reset does not effect this bit. A write to the force event register (FEVT) does not affect this bit. The reset value is effected by the external card detection pin. This bit shows the value on the SDHC_CD_B pin (that is, when a card is inserted in the socket, it is 0 on the SDHC_CD_B input, and consequently the CDS reads 1.)  0b - No card present (SDHC_CD_B =1) 1b - Card present SDHC_CD_B =0)
14 —	Reserved
15 CINS	Card inserted. This bit indicates whether a card has been inserted. The eSDHC debounces this signal so that the host driver will not need to wait for it to stabilize. Changing from a 0 to 1 generates a card insertion interrupt in the interrupt status register (IRQSTAT). Changing from a 1 to 0 generates a card removal interrupt in the interrupt status register (IRQSTAT). A write to the force event register (FEVT) does not effect this bit.  The software reset for all in the system control register (SYSCTL) does not effect this bit. A software reset does not effect this bit.  0b - Power on reset or no card 1b - Card inserted
16-19 —	Reserved
20 BREN	Buffer read enable. This status bit is used for non-DMA read transfers. The eSDHC may implement multiple buffers to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this bit is high, valid data greater than the watermark level exist in the buffer. A change of this bit from 1 to 0 occurs when any read from the buffer is made. A change of this bit from 0 to 1 occurs when there is enough valid data ready in the buffer and the buffer read ready interrupt has been generated and enabled.  0b - Read disable 1b - Read enable

Table continues on the next page...

Field	Function
21 BWEN	Buffer write enable. This status bit is used for non-DMA write transfers. The eSDHC can implement multiple buffers to transfer data efficiently. This read only flag indicates if space is available for write data. If this bit is 1, valid data greater than the watermark level can be written to the buffer. A change of this bit from 1 to 0 occurs when any write to the buffer is made. A change of this bit from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the buffer write ready interrupt is generated and enabled.  0b - Write disable 1b - Write enable
22 RTA	Read transfer active. This status bit is used for detecting completion of a read transfer. <ul style="list-style-type: none"><li>• This bit is set for either of the following conditions:<ul style="list-style-type: none"><li>• After the end bit of the read command.</li><li>• When read operation is restarted by writing a 1 to PROCTL[CREQ].</li></ul></li><li>• This bit is cleared for either of the following conditions:<ul style="list-style-type: none"><li>• When the last data block as specified by block length is transferred to the system.</li><li>• In the case of ADMA2, end of read operation is designated by descriptor table.</li><li>• When all valid data blocks in the host controller have been transferred to the system and no current block transfers are being sent as a result of the stop at block gap request being set to 1.</li></ul></li></ul> A transfer complete interrupt is generated when this bit changes to 0.  0b - No valid data 1b - Transferring data
23 WTA	Write transfer active. This status bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the eSDHC. <ul style="list-style-type: none"><li>• This bit is set in either of the following cases:<ul style="list-style-type: none"><li>• After the end bit of the write command.</li><li>• When write operation is restarted by writing a 1 to PROCTL[CREQ].</li></ul></li><li>• This bit is cleared in either of the following cases:<ul style="list-style-type: none"><li>• After getting the CRC status of the last data block as specified by the transfer count (single and multiple). In case of ADMA2, transfer count is designated by descriptor table.</li><li>• After getting the CRC status of any block where data transmission is about to be stopped by a stop at block gap request.</li></ul></li></ul> During a write transaction, a block gap event interrupt is generated when this bit is changed to 0, as result of the stop at block gap request being set. This status is useful for the host driver in determining when to issue commands during write busy state.  0b - No valid data 1b - Transferring data
24 SDOFF	SD clock gated off internally. This status bit indicates that the SD clock is internally gated off, because of buffer over/under-run or read pause without read wait assertion.  0b - SD clock is active 1b - SD clock is gated off
25-27 —	Reserved
28 SDSTB	SD clock stable. This status bit indicates that the internal card clock is stable. This bit is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear SYSCTL[SDCLKEN] to remove glitch on the card clock when the frequency is changing.  0b - Clock is changing frequency and not stable 1b - Clock is stable
29	Data line active. This status bit indicates whether one of the DAT lines on the SD bus is in use.

*Table continues on the next page...*

## eSDHC register descriptions

Field	Function
DLA	<p>In the case of read transactions:</p> <ul style="list-style-type: none"> <li>This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a block gap event interrupt in the interrupt status register (IRQSTAT).</li> <li>This bit will be set in either of the following cases: <ul style="list-style-type: none"> <li>After the end bit of the read command.</li> <li>When writing a 1 to PROCTL[CREQ] to restart a read transfer.</li> </ul> </li> <li>This bit should be cleared in either of the following cases: <ul style="list-style-type: none"> <li>When the end bit of the last data block is sent from the SD bus to the host controller. In case of ADMA2, the last block is designated by the last transfer of descriptor table.</li> <li>When a read transfer is stopped at the block gap initiated by a stop at block gap request.</li> </ul> </li> <li>The eSDHC will wait at the next block gap by driving read wait at the start of the interrupt cycle. If the read wait signal is already driven (data buffer cannot receive data), the eSDHC can wait for a current block gap by continuing to drive the read wait signal. It is necessary to support read wait in order to use the suspend/resume function. This bit will remain 1 during read wait.</li> </ul> <p>In the case of write transactions:</p> <ul style="list-style-type: none"> <li>This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a transfer complete interrupt in the interrupt status register (IRQSTAT).</li> <li>This bit will be set in either of the following cases: <ul style="list-style-type: none"> <li>After the end bit of the write command.</li> <li>When writing to 1 to PROCTL[CREQ] to continue a write transfer.</li> </ul> </li> <li>This bit will be cleared in either of the following cases: <ul style="list-style-type: none"> <li>When the SD card releases write busy of the last data block. If SD card does not drive busy signal for 8 SD Clocks, the host controller should consider the card drive "Not Busy". In case of ADMA2, the last block is designated by the last transfer of descriptor table.</li> <li>When the SD card releases write busy, prior to waiting for write transfer, and as a result of a stop at block gap request.</li> </ul> </li> </ul> <p>In the case of command with busy pending:</p> <ul style="list-style-type: none"> <li>Command with busy. This status indicates whether a command indicates busy (for example, erase command for memory) is executing on the SD bus. This bit is set after the end bit of the command with busy and cleared when busy is de-asserted.</li> </ul> <p>Changing this bit from 1 to 0 generate a transfer complete interrupt in the interrupt status register (IRQSTAT).</p> <p>0b - DAT line inactive 1b - DAT line active</p>
30 CDIHB	<p>Command inhibit (DAT). This status bit is generated if either the DAT line active or the read transfer active is set to 1. If this bit is 0, it indicates the eSDHC can issue the next SD/MMC Command. Commands with busy signal belong to command inhibit (DAT) (for example, R1b, R5b type). Changing from 1 to 0 generates a transfer complete interrupt in the interrupt status register (IRQSTAT).</p> <p><b>NOTE:</b> The SD host driver can save registers for a suspend transaction after this bit has changed from 1 to 0.</p> <p>0b - Can issue command which uses the DAT line 1b - Cannot issue command which uses the DAT line</p>
31 CIHB	<p>Command inhibit (CMD). If this status bit is 0, it indicates that the CMD line is not in use and the eSDHC can issue a SD/MMC Command using the CMD line.</p> <p>This bit is set also immediately after the transfer type register (XFERTYP) is written. This bit is cleared when the command response is received. Even if the command inhibit (DAT) is set to 1, Commands using only the CMD line can be issued if this bit is 0. Changing from 1 to 0 generates a command complete interrupt in the interrupt status register (IRQSTAT). If the eSDHC cannot issue the command</p>

Field	Function
	<p>because of a command conflict error (Refer to command CRC error) or because of a command not issued by Auto CMD12 Error, this bit will remain 1 and the command complete is not set. The status of issuing an Auto CMD12 does not show on this bit.</p> <p>0b - Can issue command using only CMD line 1b - Cannot issue command</p>

## 22.3.12 Protocol control register (PROCTL)

### 22.3.12.1 Offset

Register	Offset
PROCTL	28h

### 22.3.12.2 Function

There are three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether the eSDHC issues a suspend command or the SD card accepts the suspend command.

1. If the host driver does not issue a suspend command, the continue request should be used to restart the transfer.
2. If the host driver issues a suspend command and the SD card accepts it, a resume command should be used to restart the transfer.
3. If the host driver issues a suspend command and the SD card does not accept it, the continue request should be used to restart the transfer.

Any time a stop at block gap request stops the data transfer, the host driver should wait for a transfer complete (in the interrupt status register, IRQSTAT), before attempting to restart the transfer. When restarting the data transfer by continue request, the host driver should clear the stop at block gap request before or simultaneously.

### 22.3.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved					WECRM	WECINS	WECINT	Reserved				IABG	RWCTL	CREQ	SABGRE
W						0	0	0				0	0	0	0	Q
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved					VOLT_SEL	DMAS		CDS_S	CDTL	EMODE	Reserved	DTW		Reserved	
W						0	0	0	0	0	1	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.12.4 Fields

Field	Function
0-4 —	Reserved
5 WECRM	<p>Wakeup event enable on SD card removal</p> <p>Wakeup event enable on SD card removal. This bit enables a wakeup event, via a card removal, in the interrupt status register (IRQSTAT). FN_WUS (wake up support) in CIS of SDIO card does not effect this bit. When this bit is set, the card removal status and the eSDHC interrupt can be asserted without SDHC_CLK toggling. When the wakeup feature is not enabled, the SDHC_CLK must be active in order to assert the card removal status and the eSDHC interrupt.</p> <p>0b - Disable 1b - Enable</p>
6 WECINS	<p>Wakeup event enable on SD card insertion</p> <p>Wakeup event enable on SD card insertion. This bit enables a wakeup event, via a card insertion, in the interrupt status register (IRQSTAT). FN_WUS (wake up support) in CIS of the SDIO card does not affect this bit. When this bit is set, the card insertion status and the eSDHC interrupt can be asserted without SDHC_CLK toggling. When the wakeup feature is not enabled, the SDHC_CLK must be active in order to assert the card insertion status and the eSDHC interrupt.</p> <p>0b - Disable 1b - Enable</p>
7 WECINT	<p>Wakeup event enable on card interrupt</p> <p>Wakeup event enable on card interrupt. This bit enables a wakeup event, via a card interrupt, in the interrupt status register (IRQSTAT). This bit can be set to 1 if FN_WUS (wake up support) in CIS of SDIO card is set to 1. When this bit is set, the card interrupt status and the eSDHC interrupt can be asserted without SDHC_CLK toggling. When the wakeup feature is not enabled, the SDHC_CLK must be active in order to assert the card interrupt status and the eSDHC interrupt.</p> <p>0b - Disable 1b - Enable</p>

Table continues on the next page...

Field	Function
8-11 —	Reserved
12 IABG	Interrupt at block gap. This bit is valid only in 4-bit mode of the SDIO card, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this bit should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it should set this bit according to the CCCR of the card.  0b - Disable 1b - Enable
13 RWCTL	Read wait control. The read wait function is optional for SDIO cards. If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using the DAT[2] line. Otherwise the eSDHC has to stop the SD Clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it should set this bit according to the CCCR of the card. If the card does not support read wait, this bit should never be set to 1, otherwise DAT line conflicts may occur. If this bit is set to 0, stop at block gap during read operation is also supported, but the eSDHC will stop the SD clock to pause reading operation.  0b - Disable read wait control, and stop SD clock at block gap when SABGREQ bit is set 1b - Enable read wait control, and assert read wait without stopping SD clock at block gap when SABGREQ bit is set
14 CREQ	Continue request. This bit is used to restart a transaction, which was stopped using the stop at block gap request. To cancel stop at the block gap, set stop at block gap request to 0 and set this bit 1 to restart the transfer.  The eSDHC automatically clears this bit in either of the following cases: <ul style="list-style-type: none"><li>• In the case of a read transaction, the DAT line active changes from 0 to 1 as a read transaction restarts.</li><li>• In the case of a write transaction, the write transfer active changes from 0 to 1 as the write transaction restarts.</li></ul> Therefore, it is not necessary for host driver to set this bit to 0. If stop at block gap request is set to 1, any write to this bit is ignored.  0b - No effect 1b - Restart
15 SABGREQ	Stop at block gap request  Stop at block gap request. This bit is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver should leave this bit set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read wait is used to stop the read transaction at the block gap. The eSDHC will honor the stop at block gap request for write transfers, but for read transfers it requires that the SDIO card support read wait. Therefore, the host driver should not set this bit during read transfers unless the SDIO card supports read wait and has set the read wait control to 1, otherwise the eSDHC will stop the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the data port register, the host driver should set this bit after all block data is written. If this bit is set to 1, the host driver should not write data to the data port register after a block is sent. Once this bit is set, the host driver should not clear this bit before the transfer complete bit in (IRQSTAT) is set, otherwise the eSDHCs behavior is undefined.  This bit effects read transfer active, write transfer active, PRSSTAT[DLA] and PRSSTAT[CDIHB].  0b - Transfer 1b - Stop
16-20 —	Reserved

*Table continues on the next page...*

## eSDHC register descriptions

Field	Function
21 VOLT_SEL	Voltage selection  Voltage selection. Change the value of output signal SDHC_VS , to control the SD bus supply voltage for external card. There must be a control circuit out of eSDHC to change the voltage.  0b - Change the SD Bus Supply voltage to high voltage range, around 3.0V 1b - Change the SD bus supply voltage to low voltage range, around 1.8V
22-23 DMAS	DMA select  DMA select. This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation.  00b - Single DMA is selected 01b - ADMA1 is selected 10b - 32-bit address ADMA2 is selected 11b - Reserved
24 CDSS	Card detect signal selection. This bit selects the source for the card detection.  0b - SD CD pin is selected (for normal purposes) 1b - Card detection test level is selected (for test purposes)
25 CDTL	Card detect test level. This is bit is enabled while the card detection signal selection is set to 1 and it indicates card insertion.  0b - Card detect test level is 0, no card inserted 1b - Card detect test level is 1, card inserted
26-27 EMODE	Endian mode. The eSDHC supports little and big endian mode for transferring between buffer port register and data buffer.  00b - Big endian mode 01b - Reserved 10b - Little endian mode 11b - Reserved
28 —	Reserved
29-30 DTW	Data transfer width. This bit selects the data width of the SD bus for a data transfer. The host driver should set it to match the data width of the card. Possible Data transfer widths are 1-bit, 4-bits, and 8-bits.  00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
31 —	Reserved

### 22.3.13 System Control Register when ESDHCCTL[CRS=0] (SYSCTL\_ESDHCCTL\_CRS\_0)

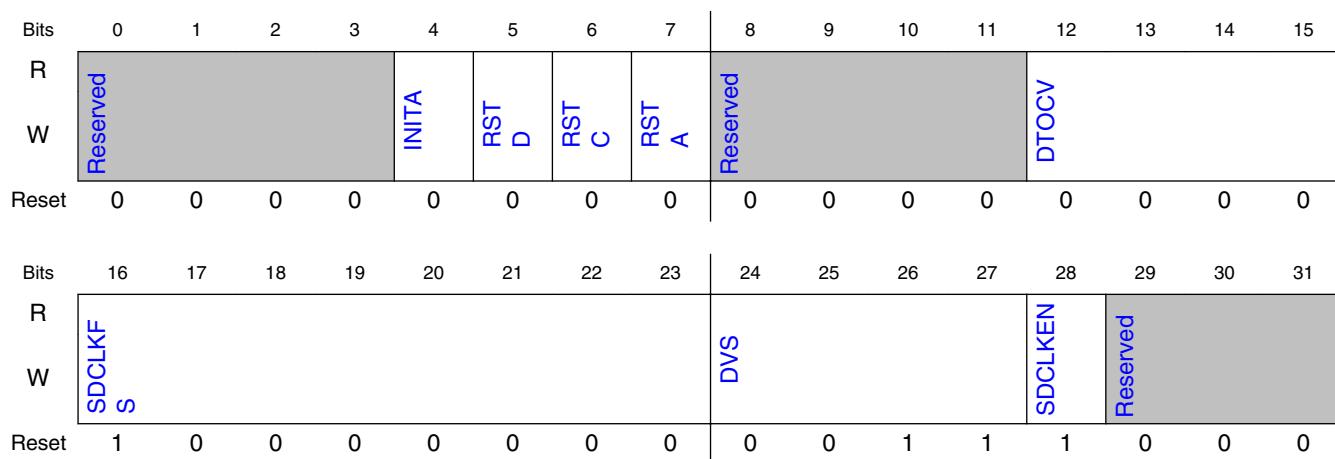
### 22.3.13.1 Offset

Register	Offset
SYSCTL_ESDHCCCTL_CRS_0	2Ch

### 22.3.13.2 Function

The clock divider mode(16-27 bits in this register) is selected according to Clock Register Select field in eSDHC Control register. Other bits of the register remain unaffected by clock register select value.

### 22.3.13.3 Diagram



### 22.3.13.4 Fields

Field	Function
0-3	Reserved
4 INITA	Initialization active. When this bit is set, 80 SD clocks are sent to the card. After the 80 clocks are sent, this bit is self cleared. This bit is very useful during the card power-up period when 74 SD-Clocks are needed. Writing 1 to this bit when this bit is already 1 has no effect. Writing 0 to this bit at any time has no effect. When either PRSSTAT[CIHB] or PRSSTAT[CDIHB] are set, writing 1 to this bit is ignored (that is, when command line or data lines are active, write to this bit is not allowed). On the other hand, when this bit is set, that is, during initialization active period, it is allowed to issue command, and the command bit

*Table continues on the next page...*

## eSDHC register descriptions

Field	Function
	stream will appear on the CMD pad after all 80 clock cycles are done. So when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs send 80 cycles to the card and does not want to wait till this bit is self cleared.
5 RSTD	<p>Software reset for DAT line. Only part of the data circuit is reset. DMA circuit is also reset.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• Data port register           <ul style="list-style-type: none"> <li>• Buffer is cleared and initialized</li> </ul> </li> <li>• Present state register (PRSSTAT)           <ul style="list-style-type: none"> <li>• Buffer read enable</li> <li>• Buffer write enable</li> <li>• Read transfer active</li> <li>• Write transfer active</li> <li>• DAT line active</li> <li>• Command inhibit (DAT)</li> </ul> </li> <li>• Protocol control register (PROCTL)           <ul style="list-style-type: none"> <li>• Continue request</li> <li>• Stop at block gap request</li> </ul> </li> <li>• Interrupt status register (IRQSTAT)           <ul style="list-style-type: none"> <li>• Buffer read ready</li> <li>• Buffer write ready</li> <li>• DMA interrupt</li> <li>• Block gap event</li> <li>• Transfer complete</li> </ul> </li> </ul> <p><b>NOTE:</b> This bit will be self cleared by eSDHC when Software Reset for Data is complete, so Software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
6 RSTC	<p>Software reset for CMD line. Only part of the command circuit is reset.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• PRSSTAT[CIHB]</li> <li>• IRQSTAT[CC]</li> </ul> <p><b>NOTE:</b> This bit will be self cleared by eSDHC when Software Reset for Command is complete, so software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
7 RSTA	<p>Software reset for all. This reset effects the entire host controller except for the card detection circuit. Register bits of type R, RW, RW1C are cleared. During its initialization, the host driver should set this bit to 1 to reset the eSDHC. The eSDHC should reset this bit to 0 when the capabilities registers are valid and the host driver can read them. Additional use of software reset for all does not affect the value of the capabilities registers. After this bit is set, it is recommended that the host driver reset the external card and re-initialize it.</p> <p><b>NOTE:</b> The Software Reset For All in the System Control register clears Card Insertion and Card Removal bits in Interrupt Status Register. Software should issue partial reset(Reset for Command and Reset for Data) instead of Reset for All, if it wants to reset eSDHC without clearing these Interrupt Status Register bits.</p> <p>This bit will be self cleared by eSDHC when Software Reset for All is complete, so Software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
8-11	Reserved

Table continues on the next page...

Field	Function
—	
12-15 DTOCV	<p>Data timeout counter value</p> <p>Data timeout counter value. This value determines the interval by which DAT line timeouts are detected. Refer to the data timeout error bit in the <a href="#">Interrupt status enable register (IRQSTATEN)</a> for information on factors that dictate time-out generation. Time-out clock frequency will be generated by dividing the SDCLK value by this value.</p> <p>The host driver can clear IRQSTATEN[DTOESEN] to prevent inadvertent time-out events.</p> <ul style="list-style-type: none"> <li>0000b - SDCLK x 2<sup>13</sup></li> <li>0001b - SDCLK x 2<sup>14</sup></li> <li>1110b - SDCLK x 2<sup>27</sup></li> <li>1111b - Reserved</li> </ul>
16-23 SDCLKFS	<p>SDCLK frequency select. This register is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this register holds the prescaler (this register) and divisor (next register) of the base clock frequency register.</p> <p>Base clock can be selected by programming ESDHCCTL[PCS]. It selects between platform clock and peripheral clock / 2 .</p> <p>Setting 0x00 bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of the base clock and the following divisor bits.</p> <p>The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor)</p> <p>For example, if the base clock frequency is 96 MHz and the target frequency is 25 MHz, then choosing the prescaler value of 0x01 and divisor value of 0x1 will yield 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 KHz, the prescaler value of 0x08 and divisor value of 0xE yields the exact clock value of 400 KHz.</p> <p>The reset value of this bit field is 0x80, so if the input base clock is about 96 MHz, the default SD clock after reset is 375 KHz.</p> <p>The programmed SD Clock frequency shall never exceed maximum SD clock supported by the card.</p> <p><b>NOTE:</b> Both DVS and SDCLKFS fields should not be programmed 0 simultaneously.</p> <p>Only the following settings are allowed:</p> <ul style="list-style-type: none"> <li>00000000b - Base clock</li> <li>00000001b - Base clock divided by 2</li> <li>00000010b - Base clock divided by 4</li> <li>00000100b - Base clock divided by 8</li> <li>00001000b - Base clock divided by 16</li> <li>00100000b - Base clock divided by 32</li> <li>01000000b - Base clock divided by 64</li> <li>10000000b - Base clock divided by 128</li> <li>10000000b - Base clock divided by 256</li> </ul>
24-27 DVS	<p>Divisor. This register is used to provide a more exact divisor to generate the desired SD clock frequency.</p> <p>Note the divisor can even support odd divisor without deterioration of duty cycle. The settings are as following:</p> <ul style="list-style-type: none"> <li>0000b - Divisor by 1</li> <li>0001b - Divisor by 2</li> <li>1110b - Divisor by 15</li> <li>1111b - Divisor by 16</li> </ul>

Table continues on the next page...

## eSDHC register descriptions

Field	Function
28 SDCLKEN	SD clock enable. the host controller should stop SDCLK when writing this bit to 0. SDCLK frequency can be changed when this bit is 0. Then, the host controller should maintain the same clock frequency until SDCLK is stopped (stop at SDCLK=0). If PRSSTAT[CINS] is cleared, this bit should be cleared by the host driver to save power.
29-31 —	Reserved

## 22.3.14 System Control Register when ESDHCCTL[CRS=1] (SYSCTL\_ESDHCCCTL\_CRS\_1)

### 22.3.14.1 Offset

Register	Offset
SYSCTL_ESDHCCCTL_CRS_1	2Ch

### 22.3.14.2 Function

System Control Register when ESDHCCTL[CRS=1]

### 22.3.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved				INITA	RST D	RST C	RST A	Reserved				DTOCV			
W					0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SDCLKF S								USDCLKF S		CG S	Reserved	SDCLKEN	Reserved		
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 22.3.14.4 Fields

Field	Function
0-3 —	Reserved
4 INITA	Initialization active. When this bit is set, 80 SD clocks are sent to the card. After the 80 clocks are sent, this bit is self cleared. This bit is very useful during the card power-up period when 74 SD-Clocks are needed. Writing 1 to this bit when this bit is already 1 has no effect. Writing 0 to this bit at any time has no effect. When either PRSSTAT[CIHB] or PRSSTAT[CDIHB] are set, writing 1 to this bit is ignored (that is, when command line or data lines are active, write to this bit is not allowed). On the other hand, when this bit is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream will appear on the CMD pad after all 80 clock cycles are done. So when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs send 80 cycles to the card and does not want to wait till this bit is self cleared.
5 RSTD	<p>Software reset for DAT line. Only part of the data circuit is reset. DMA circuit is also reset.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• Data port register           <ul style="list-style-type: none"> <li>• Buffer is cleared and initialized</li> </ul> </li> <li>• Present state register (PRSSTAT)           <ul style="list-style-type: none"> <li>• Buffer read enable</li> <li>• Buffer write enable</li> <li>• Read transfer active</li> <li>• Write transfer active</li> <li>• DAT line active</li> <li>• Command inhibit (DAT)</li> </ul> </li> <li>• Protocol control register (PROCTL)           <ul style="list-style-type: none"> <li>• Continue request</li> <li>• Stop at block gap request</li> </ul> </li> <li>• Interrupt status register (IRQSTAT)           <ul style="list-style-type: none"> <li>• Buffer read ready</li> <li>• Buffer write ready</li> <li>• DMA interrupt</li> <li>• Block gap event</li> <li>• Transfer complete</li> </ul> </li> </ul> <p><b>NOTE:</b> This bit will be self cleared by eSDHC when Software Reset for Data is complete, so Software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
6 RSTC	<p>Software reset for CMD line. Only part of the command circuit is reset.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• PRSSTAT[CIHB]</li> <li>• IRQSTAT[CC]</li> </ul> <p><b>NOTE:</b> This bit will be self cleared by eSDHC when Software Reset for Command is complete, so software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
7 RSTA	Software reset for all. This reset effects the entire host controller except for the card detection circuit. Register bits of type R, RW, RW1C are cleared. During its initialization, the host driver should set this bit to 1 to reset the eSDHC. The eSDHC should reset this bit to 0 when the capabilities registers are valid

*Table continues on the next page...*

## eSDHC register descriptions

Field	Function
	<p>and the host driver can read them. Additional use of software reset for all does not affect the value of the capabilities registers. After this bit is set, it is recommended that the host driver reset the external card and re-initialize it.</p> <p><b>NOTE:</b> The software reset for all in the system control register clears card insertion and card removal bits in interrupt status register. Software should issue partial reset (reset for command and reset for data) instead of reset for all, if it wants to reset eSDHC without clearing these interrupt status register bits.</p> <p>This bit will be self cleared by eSDHC when software reset for all is complete, so software should poll for it to be cleared after setting it.</p> <p>0b - No reset 1b - Reset</p>
8-11 —	Reserved
12-15 DTOCV	<p>Data timeout counter value</p> <p>Data timeout counter value. This value determines the interval by which DAT line timeouts are detected. Refer to the data timeout error bit in the <a href="#">Interrupt status enable register (IRQSTATEN)</a> for information on factors that dictate time-out generation. Time-out clock frequency will be generated by dividing the SDCLK value by this value.</p> <p>The host driver can clear IRQSTATEN[DTOESEN] to prevent inadvertent time-out events.</p> <p>0000b - SDCLK x 2<sup>13</sup> 0001b - SDCLK x 2<sup>14</sup> 1110b - SDCLK x 2<sup>27</sup> 1111b - Reserved</p>
16-23 SDCLKFS	<p>SDCLK Frequency Select:</p> <p>This register is used to select the frequency of the SDCLK pin. 10-bit divisor value is formed by concatenating USDCLKFS(2-bit) and SDCLKFS(8-bit), that is, Divisor = {USDCLKFS[0:1], SDCLKFS[0:7]}</p> <p>Following Divisor definition is selected depending on Clock Generation Select value:</p> <p><b>10-bit Divided Clock Mode</b></p> <p>0x3FF Base clock divided by 2048 0x04 Base clock divided by 8 0x03 Base clock divided by 6 0x02 Base clock divided by 4 0x01 Base clock divided by 2 0x00 Reserved</p> <p><b>10-bit Programmable Clock Mode</b></p> <p>0x3FF Base clock divided by 1024 0x04 Base clock divided by 5 0x03 Base clock divided by 4 0x02 Base clock divided by 3 0x01 Base clock divided by 2 0x00 Reserved</p> <p>The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (divisor)</p>

Table continues on the next page...

Field	Function
24-25 USDCLKFS	Upper bits of SDCLK frequency select. This field is used to expand SDCLKFS to 10 bits. These two bits occupies most significant portion of 10-bit SDCLKFS.
26 CGS	Clock Generator Select. This field selects 10-bit SDCLKFS clock mode. 0b - Divided clock mode is selected 1b - Programmable clock mode is selected
27 —	Reserved
28 SDCLKEN	SD clock enable. The host controller shall stop SDCLK when writing this bit to 0. SDCLK frequency can be changed when this bit is 0. Then, the host controller shall maintain the same clock frequency until SDCLK is stopped (Stop at SDCLK=0). If the card inserted in the present state register is cleared, this bit should be cleared by the host driver to save power.
29-31 —	Reserved

## 22.3.15 Interrupt status register (IRQSTAT)

### 22.3.15.1 Offset

Register	Offset
IRQSTAT	30h

### 22.3.15.2 Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status bits is set to 1. For most bits, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition, otherwise the CINT bit will be asserted again.

**Table 22-5. eSDHC Status for Command Timeout Error/Command Complete Bit Combinations**

Command Complete	Command Timeout Error	Meaning of the Status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

## eSDHC register descriptions

**Table 22-6. eSDHC Status for Data Timeout Error/Transfer Complete Bit Combinations**

Transfer Complete	Data Timeout Error	Meaning of the Status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data Transfer Complete

**Table 22-7. eSDHC Status for Command CRC Error/Command Timeout Error Bit Combinations**

Command CRC Error		Command Timeout Error							Meaning of the Status						
0		0							No error						
0		1							Response Timeout Error						
1		0							Response CRC Error						
1		1							CMD line conflict						

### 22.3.15.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		RTOE	DMAE	Reserved		TNE	ADMAE	AC12E	Reserved		DEB_E	DCE	DTOE	C1_E	CEB_E
W	W1C		W1C	W1C	Reserved		W1C	W1C	W1C	Reserved		W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			RT_E	Reserved			CINT	CRM	CINS	BR_R	BWR	DINT	BG_E	TC	CC
W	W1C			W1C	Reserved			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.15.4 Fields

Field	Function
0-1	Reserved
—	
2	Register access timeout error. This bit indicate that register access has timed-out. 0b - No timeout error

Table continues on the next page...

Field	Function
RTOE	1b - Timeout error
3 DMAE	DMA error DMA error. This bit indicates that DMA (SDMA or ADMA) transfer has failed. 0b - No error 1b - Error
4 —	Reserved
5 TNE	Tuning Error. This bit is set when an unrecoverable error is detected in a tuning circuit except during tuning procedure (Occurrence of an error during tuning procedure is indicated by Sampling Clock Select). By detecting Tuning Error, Host Driver needs to abort a command executing and perform tuning. The Tuning Error is higher priority than the other error interrupts generated during data transfer. By detecting Tuning Error, the Host Driver should discard data transferred by a current read/write command and retry data transfer after the Host Controller retrieved from tuning circuit error. This bit might not be set in some cases, but SD command or Data error might be set; Driver should consider it as tuning circuit error and perform tuning procedure. 0b - No error 1b - Error
6 ADMAE	ADMA error. This bit is set when the host controller detects errors during ADMA operation. The state of the ADMA at an error occurrence is saved in the ADMA error status register. 0b - No error 1b - Error
7 AC12E	Auto CMD12 error. Occurs when detecting that one of the bits in the Auto CMD12 error status register (AUTOC12ERR) has changed from 0 to 1. This bit is set to 1, not only when the errors in Auto CMD12 occur, but also when the Auto CMD12 is not executed due to the previous command error. 0b - No error 1b - Error
8 —	Reserved
9 DEBE	Data end bit error. Occurs either when detecting 0 at the end bit position of read data, which uses the DAT line, or at the end bit position of the CRC. 0b - No error 1b - Error
10 DCE	Data CRC error. Occurs when detecting a CRC error when transferring read data, which uses the DAT line, or when detecting the write CRC status having a value other than 010. 0b - No error 1b - Error
11 DTOE	Data timeout error. Occurs when detecting one of following time-out conditions. <ul style="list-style-type: none"> <li>• Busy time-out for R1b, R5b type</li> <li>• Busy time-out after write CRC status</li> <li>• Write CRC status time-out</li> <li>• Read data time-out</li> </ul> 0b - No error 1b - Time out
12 CIE	Command index error. Occurs if a command index error occurs in the command response. 0b - No error 1b - Error
13	Command end bit error. Occurs when detecting that the end bit of a command response is 0.

*Table continues on the next page...*

## eSDHC register descriptions

Field	Function
CEBE	0b - No error 1b - End bit error generated
14 CCE	Command CRC error. A command crc error is generated in two cases: <ul style="list-style-type: none"><li>• If a response is returned and the command timeout error is set to 0 (indicating no time-out), this bit is set when detecting a CRC error in the command response.</li><li>• The eSDHC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the eSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then the eSDHC should abort the command (stop driving CMD line) and set this bit to 1. The command timeout error should also be set to 1 to distinguish CMD line conflict.</li></ul> 0b - No error 1b - CRC error generated
15 CTOE	Command timeout error  Command timeout error. Occurs only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the eSDHC detects a CMD line conflict, in which case a command CRC error should also be set (as shown in ), this bit should be set without waiting for 64 SDCLK cycles. This is because the command will be aborted by the eSDHC.  0b - No error 1b - Time out
16-18 —	Reserved
19 RTE	Re-tuning event. This status is set if re-tuning request in the eSDHC control register changes from 0 to 1. eSDHC requests host driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning. 0b - Re-tuning is not required 1b - Re-tuning should be performed
20-22 —	Reserved
23 CINT	Card interrupt. Writing this bit to 1 does not clear this bit. It is cleared by resetting the SD card interrupt factor. In 1-bit mode, the host controller should detect the card interrupt without SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the SD card and the interrupt to the host system. It is necessary to define how to handle this delay.  When this status has been set and the host driver needs to start this interrupt service, IRQSTATEN[CINTSEN] should be set to 0 in order to clear the card interrupt statuses latched in the eSDHC and to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset interrupt factors in the SD card and the interrupt signal may not be asserted), set the card interrupt status enable bit to 1 and start sampling the interrupt signal again.  0b - No card interrupt 1b - Generate card interrupt
24 CRM	Card removal. This status is set if the PRSSTAT[CINS] changes from 1 to 0.  When the host driver writes this bit to 1 to clear this status, the status of the PRSSTAT[CINS] should be confirmed. Because the card detect state may possibly be changed when the host driver clear this bit and interrupt event may not be generated.  <b>NOTE:</b> The Software Reset For All in the System Control register clears this bit. Software should issue partial reset(Reset for Command and Reset for Data) instead of Reset for All, if it wants to reset eSDHC without clearing this bit.  eSDHC does not implement de-bouncing circuit on card detect pin, so Software should check Card Inserted in Present State register to confirm card insertion/removal status. 0b - Card state unstable or inserted

Table continues on the next page...

Field	Function
	1b - Card removed
25 CINS	<p>Card insertion. This status is set if PRSSTAT[CINS] changes from 0 to 1.</p> <p>When the host driver writes this bit to 1 to clear this status, the status of the PRSSTAT[CINS] should be confirmed. Because the card detect state may possibly be changed when the host driver clear this bit and interrupt event may not be generated.</p> <p><b>NOTE:</b> The Software Reset For All in the System Control register clears this bit. Software should issue partial reset(Reset for Command and Reset for Data) instead of Reset for All, if it wants to reset eSDHC without clearing this bit.</p> <p>eSDHC does not implement de-bouncing circuit on card detect pin, so Software should check Card Inserted in Present State register to confirm card insertion/removal status.</p> <p>0b - Card state unstable or removed 1b - Card inserted</p>
26 BRR	<p>Buffer read ready</p> <p>Buffer read ready. This status bit is set if PRSSTAT[BREN] changes from 0 to 1. Refer to the description of the buffer read enable bit in <a href="#">Present state register (PRSSTAT)</a> for additional information.</p> <p>0b - Not ready to read buffer 1b - Ready to read buffer</p>
27 BWR	<p>Buffer write ready</p> <p>Buffer write ready. This status bit is set if the PRSSTAT[BWEN] changes from 0 to 1. Refer to the description of the buffer write enable bit in <a href="#">Present state register (PRSSTAT)</a> for additional information.</p> <p>0b - Not ready to write buffer 1b - Ready to write buffer</p>
28 DINT	<p>DMA interrupt</p> <p>DMA interrupt. Occurs only when the DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this bit will not be set. Instead, the DMAE bit will be set. Either single DMA or ADMA finishes data transferring, this bit will be set.</p> <p>0b - No DMA Interrupt 1b - DMA Interrupt is generated</p>
29 BGE	<p>Block gap event. If PROCTL[SABGREQ] is set, this bit is set when a read or write transaction is stopped at a block gap. If PROCTL[SABGREQ] is not set to 1, this bit is not set to 1.</p> <ul style="list-style-type: none"> <li>In the case of a read transaction, this bit is set at the falling edge of the DAT Line active status (when the transaction is stopped at SD bus timing). The read wait must be supported in order to use this function.</li> <li>In the case of a write transaction, this bit is set at the falling edge of write transfer active status (after getting CRC status at SD Bus timing).</li> </ul> <p>0b - No block gap event 1b - Transaction stopped at block gap</p>
30 TC	<p>Transfer complete. This bit is set when a read/write transfer and a command with busy is completed.</p> <p>While performing tuning procedure (Execute Tuning is set to 1), Transfer Complete is not set for CMD19 execution.</p> <p>0b - Transfer not complete 1b - Transfer complete</p>
31 CC	<p>Command complete. This bit is set when the end bit of the command response is received (except Auto CMD12). Refer to PRSSTAT[CIHB].</p> <p>0b - Command not complete 1b - Command complete</p>

## 22.3.16 Interrupt status enable register (IRQSTATEN)

### 22.3.16.1 Offset

Register	Offset
IRQSTATEN	34h

### 22.3.16.2 Function

Setting the bits to 1 in the IRQSTATEN enables the corresponding interrupt status to be set by the specified event. If any bit is cleared, the corresponding interrupt status bit is also cleared (that is, when the bit in this register is cleared, the corresponding bit in interrupt status register, IRQSTAT, is always 0).

#### NOTE

- Depending on how PROCTL[IABG] is set, eSDHC may be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There will be some delays on the card interrupt, asserted from the card, to the time the host system is informed.
- To detect a CMD line conflict, the host driver must set both IRSTAT[CTOESEN] and IRSTAT[CCESEN] to 1.

### 22.3.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		RTOESE N	DMAESEN	Reserved	TNESE N	ADMAESEN	AC12ESEN	Reserved	DEBESE N	DCESE N	DTOESE N	CIESE N	CEBESE N	CCESE N	CTOESE N
W	0	0	1	1	0	1	1	1	0	1	1	1	1	1	1	1
Reset	0	0	0	1	1	0	0	1	0	1	1	1	1	1	1	1

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		RTESE N	Reserved			CINTSEN	CRMSEN	CINSEN	BRRSE N	BWRSE N	DINTSEN	BGESE N	TCSE N	CCSE N	
W	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1
Reset	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1

### 22.3.16.4 Fields

Field	Function
0-1 —	Reserved
2 RTOESEN	Register access timeout status enable 0b - Masked 1b - Enabled
3 DMAESEN	DMA error status enable. 0b - Masked 1b - Enabled
4 —	Reserved
5 TNESEN	Tuning error status enable 0b - Masked 1b - Enabled
6 ADMAESEN	ADMA error status enable. 0b - Masked 1b - Enabled
7 AC12ESEN	Auto CMD12 error status enable. 0b - Masked 1b - Enabled
8 —	Reserved
9 DEBESEN	Data end bit error status enable. 0b - Masked 1b - Enabled
10 DCESEN	Data CRC error status enable. 0b - Masked 1b - Enabled
11 DTOESEN	Data timeout error status enable. 0b - Masked 1b - Enabled
12 CIESEN	Command index error status enable. 0b - Masked 1b - Enabled
13 CEBESEN	Command end bit error status enable. 0b - Masked 1b - Enabled
14 CCESSEN	Command CRC error status enable. 0b - Masked 1b - Enabled

Table continues on the next page...

## eSDHC register descriptions

Field	Function
15 CTOESEN	Command timeout error status enable. 0b - Masked 1b - Enabled
16-18 —	Reserved
19 RTESEN	Re-tuning event status enable. 0b - Masked 1b - Enabled
20-22 —	Reserved
23 CINTSEN	Card interrupt status enable. If this bit is set to 0, the eSDHC will clear the interrupt request to the system. The card interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The host driver should clear the card interrupt status enable before servicing the card interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
24 CRMSEN	Card removal status enable. 0b - Masked 1b - Enabled
25 CINSEN	Card insertion status enable. 0b - Masked 1b - Enabled
26 BRRSEN	Buffer read ready status enable. 0b - Masked 1b - Enabled
27 BWRSEN	Buffer write ready status enable. 0b - Masked 1b - Enabled
28 DINTSEN	DMA interrupt status enable. 0b - Masked 1b - Enabled
29 BGESEN	Block gap event status enable. 0b - Masked 1b - Enabled
30 TCSEN	Transfer complete status enable. 0b - Masked 1b - Enabled
31 CCSEN	Command complete status enable. 0b - Masked 1b - Enabled

## 22.3.17 Interrupt signal enable register (IRQSIGEN)

### 22.3.17.1 Offset

Register	Offset
IRQSIGEN	38h

### 22.3.17.2 Function

The IRQSIGEN is used to select which interrupt status is indicated to the host system as the interrupt. These status bits all share the same interrupt line. Setting any of these bits to 1 enables interrupt generation. The corresponding status register bit will generate an interrupt when the corresponding interrupt signal enable bit is set.

### 22.3.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		RTEIE N	DMAEEN	Reserved	TNEIE N	ADMAEEN	AC12EEN	Reserved	DEBEIE N	DCEIEN	DTOEIN	CIEIE N	CEBEIE N	CCEIEN	CTOEIEN
W	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			RTEIE N	Reserved		CINTIEN		CRMEN	CINSIEN	BRRIE N	BWRIEN	DINTIEN	BGEIE N	TCIEN	CCIEN
W	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.17.4 Fields

Field	Function
0-1	Reserved
—	
2	Register timeout error interrupt enable 0b - Masked

Table continues on the next page...

## eSDHC register descriptions

Field	Function
RTOEIEN	1b - Enable
3 DMAEIEN	DMA error interrupt enable. 0b - Masked 1b - Enabled
4 —	Reserved
5 TNEIEN	Tuning error interrupt enable 0b - Masked 1b - Enable
6 ADMAEIN	ADMA Error Interrupt Enable.
7 AC12EIEN	Auto CMD12 error interrupt enable. 0b - Masked 1b - Enabled
8 —	Reserved
9 DEBEIEN	Data end bit error interrupt enable. 0b - Masked 1b - Enabled
10 DCEIEN	Data CRC error interrupt enable. 0b - Masked 1b - Enabled
11 DTOEIEN	Data timeout error interrupt enable. 0b - Masked 1b - Enabled
12 CIEIEN	Command index error interrupt enable. 0b - Masked 1b - Enabled
13 CEBEIEN	Command end bit error interrupt enable. 0b - Masked 1b - Enabled
14 CCEIEN	Command CRC error interrupt enable. 0b - Masked 1b - Enabled
15 CTOEIEN	Command timeout error interrupt enable. 0b - Masked 1b - Enabled
16-18 —	Reserved
19 RTEIEN	Re-tuning event interrupt enable 0b - Masked 1b - Enabled
20-22 —	Reserved

Table continues on the next page...

Field	Function
23 CINTIEN	Card interrupt interrupt enable. 0b - Masked 1b - Enabled
24 CRMIEN	Card removal interrupt enable. 0b - Masked 1b - Enabled
25 CINSIEN	Card insertion interrupt enable. 0b - Masked 1b - Enabled
26 BRRIEN	Buffer read ready interrupt enable. 0b - Masked 1b - Enabled
27 BWRIEN	Buffer write ready interrupt enable. 0b - Masked 1b - Enabled
28 DINTIEN	DMA interrupt enable. 0b - Masked 1b - Enabled
29 BGEIEN	Block gap event interrupt enable. 0b - Masked 1b - Enabled
30 TCIEN	Transfer complete interrupt enable. 0b - Masked 1b - Enabled
31 CCIEN	Command complete interrupt enable. 0b - Masked 1b - Enabled

## 22.3.18 Auto CMD Error Status Register / System Control 2 Register (AUTOCERR\_SYSCTL2)

### 22.3.18.1 Offset

Register	Offset
AUTOCERR_SYSCTL2	3Ch

## 22.3.18.2 Function

When the Auto CMD12 error status bit in the AUTOC12ERR is set, the host driver checks this register to identify the kind of error indicated by the Auto CMD12. This register is valid only when the auto CMD12 error status bit is set.

**Table 22-8. Relationship Between Command CRC Error and Command Timeout Error for Auto CMD12**

Auto CMD12 CRC Error	Auto CMD12 Timeout Error	Type of Error
0	0	No Error
0	1	Response Timeout Error
1	0	Response CRC Error
1	1	CMD line conflict

Changes in Auto CMD12 error status register (AUTOC12ERR) can be classified in three scenarios:

1. When the eSDHC is going to issue an Auto CMD12
  - Set bit 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set bit 0 to 0 if the Auto CMD12 is issued
2. At the end bit of an Auto CMD12 response
  - Check errors correspond to bits 1-4
  - Set bits 1-4 corresponding to detected errors
  - Clear bits 1-4 corresponding to detected errors
3. Before reading the Auto CMD12 error status bit 7
  - Set bit 7 to 1 if there is a command that cannot be issued
  - Clear bit 7 if there is no command to issue

The timing for generating the Auto CMD12 error and writing to the command register are asynchronous. After that, bit 7 should be sampled when the driver is not writing to the command register. So it is suggested to read this register only when IRQSTAT[AC12E] is set. An Auto CMD12 error interrupt is generated when one of the error bits (0-4) is set to 1. The command not issued by Auto CMD12 Error does not generate an interrupt.

### 22.3.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		AIE		Reserved				SMPCLKSEL							
W									L	EXTN						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								CNIBAC12E		Reserved		AC12IE		AC12EBE	
W									0	0	0	0	0	0	AC12CE	AC12TOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.18.4 Fields

Field	Function
0	Reserved
—	
1 AIE	Asynchronous Interrupt Enable. This bit can be set to 1 if a card supports asynchronous interrupts and asynchronous interrupt support is set to 1 in capability register. If this bit is set to 1, host driver can stop SDCLK during asynchronous interrupt period to save power. During this period, eSDHC continues to deliver the card interrupt to the system when it is asserted by card.  0b - Disabled 1b - Enabled
2-7	Reserved
—	
8 SMPCLKSEL	Sampling clock select. This bit is set by eSDHC during tuning procedure and valid after the completion of tuning (when execute tuning is cleared). Setting 1 by eSDHC means that tuning is completed successfully and setting 0 means that tuning is failed. Host driver should not write to this bit. Change of this bit is not allowed while eSDHC is receiving response or a read data block.  0b - Tuning procedure unsuccessful 1b - Tuning procedure completed successfully
9 EXTN	Execute Tuning. This bit is set to 1 by host driver to start tuning procedure and automatically cleared by eSDHC when tuning procedure is completed. The result of tuning is indicated to sampling clock select field. Tuning procedure is aborted by writing 0.  0b - Not tuned or tuning not completed 1b - Execute tuning
10-12	Reserved
—	

Table continues on the next page...

## eSDHC register descriptions

Field	Function
13-15 UHSM	<p>UHS mode select</p> <p>UHS mode select. This field is used to select one of UHS-1 modes for SD 3.0 card; and select HS200 or DDR mode for MMC card.</p> <p>Host driver should reset SDCLKEN in system control register before changing this field, and then set SDCLKEN again.</p> <ul style="list-style-type: none"> <li>000b - SDR12 for SD, or max 52 MHz mode for SD 2.0 / MMC 4.2 or older spec</li> <li>001b - SDR25 for SD</li> <li>010b - SDR50 for SD</li> <li>011b - SDR104 for SD, HS200 for MMC</li> <li>100b - DDR</li> <li>101b - Rest all the fields are reserved</li> </ul>
16-23 —	Reserved
24 CNIBAC12E	<p>Command not issued by Auto CMD12 error. Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register.</p> <ul style="list-style-type: none"> <li>0b - No error</li> <li>1b - Not Issued</li> </ul>
25-26 —	Reserved
27 AC12IE	<p>Auto CMD index error. Occurs if the command index error occurs in response to a command.</p> <ul style="list-style-type: none"> <li>0b - No error</li> <li>1b - Error, the CMD index in response is not CMD12</li> </ul>
28 AC12EBE	<p>Auto CMD end bit error. Occurs when detecting that the end bit of command response is 0 which should be 1.</p> <ul style="list-style-type: none"> <li>0b - No error</li> <li>1b - End bit error generated</li> </ul>
29 AC12CE	<p>Auto CMD CRC error. Occurs when detecting a CRC error in the command response.</p> <ul style="list-style-type: none"> <li>0b - No CRC error</li> <li>1b - CRC error met in Auto CMD12 response</li> </ul>
30 AC12TOE	<p>Auto CMD timeout error. Occurs if no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (2-4) have no meaning.</p> <ul style="list-style-type: none"> <li>0b - No error</li> <li>1b - Time out</li> </ul>
31 AC12NE	<p>Auto CMD12 not executed. If memory multiple block data transfer is not started, due to a command error, this bit is not set because it is not necessary to issue an Auto CMD12. Setting this bit to 1 means the eSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (1-4) have no meaning.</p> <p>This bit is set to 0 when Auto CMD Error is generated by Auto CMD23.</p> <ul style="list-style-type: none"> <li>0b - Executed</li> <li>1b - Not executed</li> </ul>

## 22.3.19 Host controller capabilities register (HOSTCAPBLT)

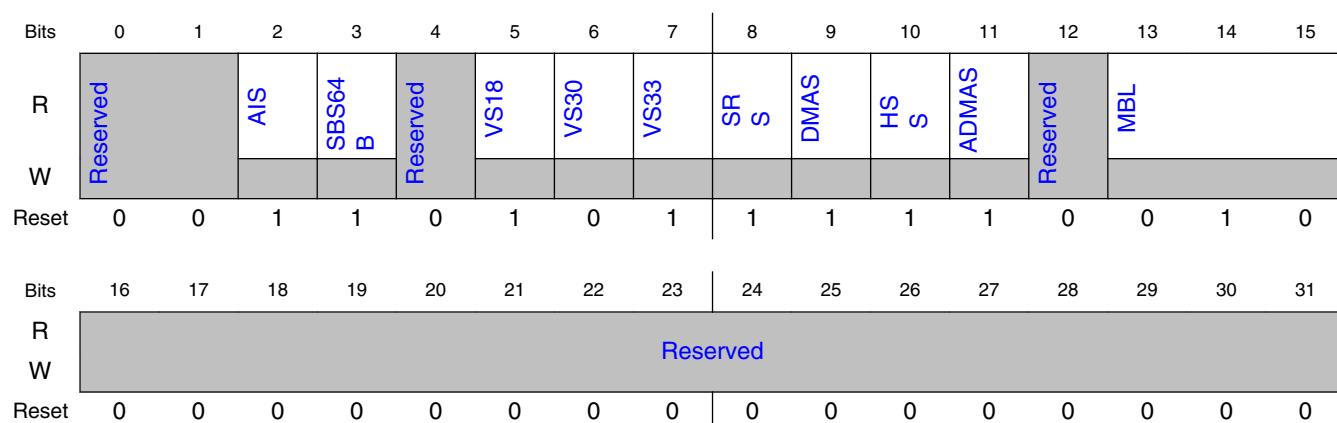
### 22.3.19.1 Offset

Register	Offset
HOSTCAPBLT	40h

### 22.3.19.2 Function

The HOSTCAPBLT provides the host driver with information specific to the eSDHC implementation. The value in this register is the power-on-reset value, and does not change with a software reset. Any write to this register is ignored.

### 22.3.19.3 Diagram



### 22.3.19.4 Fields

Field	Function
0-1	Reserved
2 AIS	Asynchronous Interrupt Support. This bit indicates whether the eSDHC supports SDIO asynchronous interrupt. Refer to SDIO Specification Version 3.0 0b - Asynchronous interrupt not supported 1b - Asynchronous interrupt supported
3 SBS64B	64-bit system bus support. This bit indicates that system supports 64-bit address descriptor mode and is connected to 64-bit address system bus. 0b - 64-bit system bus not supported 1b - 64-bit system bus supported

Table continues on the next page...

## eSDHC register descriptions

Field	Function
4 —	Reserved
5 VS18	Voltage support 1.8V. This bit should depend on the host system ability. 0b - 1.8V not supported 1b - 1.8V supported
6 VS30	Voltage Support 3.0V. This bit shall depend on the Host System ability. 0b - 3.0V not supported 1b - 3.0V supported
7 VS33	Voltage support 3.3V. This bit should depend on the host system ability. 0b - 3.3V not supported 1b - 3.3V supported
8 SRS	Suspend/resume support. This bit indicates whether the eSDHC supports suspend/resume functionality. If this bit is 0, the suspend and resume mechanism, as well as the read wait, are not supported, and the host driver should not issue either suspend or resume commands. 0b - Not supported 1b - Supported
9 DMAS	DMA support. This bit indicates whether the eSDHC is capable of using the DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
10 HSS	High speed support. This bit indicates whether the eSDHC supports high speed mode and the host system can supply a SD clock frequency from 25-50 MHz. 0b - High speed not supported 1b - High speed supported
11 ADMAS	ADMA support. This bit indicates whether the eSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
12 —	Reserved
13-15 MBL	Maximum block length. This value indicates the maximum block size that the host driver can read and write to the buffer in the eSDHC. The buffer should transfer block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011-111b - Reserved
16-31 —	Reserved

## 22.3.20 Watermark level register (WML)

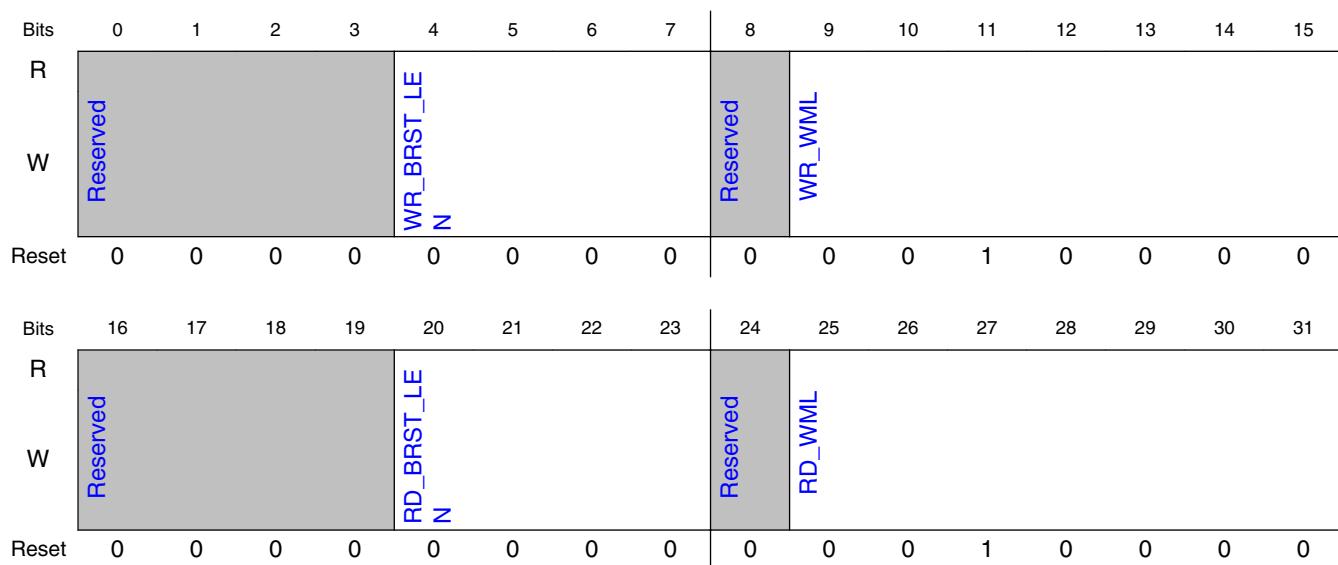
### 22.3.20.1 Offset

Register	Offset
WML	44h

### 22.3.20.2 Function

Both write and read watermark levels (FIFO threshold) are configurable in the WML. They can range from 1- 128 words. Both write and read burst lengths are also configurable. They can range from 1- 16 beats.

### 22.3.20.3 Diagram



### 22.3.20.4 Fields

Field	Function
0-3	Reserved
4-7 WR_BRST_LEN	Max write burst length. Burst length desirable for write on system bus when DMA is used. This is the maximum burst length, actual burst length may be less than this depending on other factors, such as block boundary

Table continues on the next page...

## eSDHC register descriptions

Field	Function
	0000b - 16 transfers in a single burst 0001b - 1 transfer in a single burst 0010b - 2 transfers in a single burst 1111b - 15 transfers in a single burst
8 —	Reserved
9-15 WR_WML	Write watermark level. The number of words (32-bit) used as the watermark level (FIFO threshold) for SD write in CPU polling mode.  0000000b - 128 words 0000001b - 1 word 0000010b - 2 words 1111111b - 127 words
16-19 —	Reserved
20-23 RD_BRST_LEN	Max read burst length. Burst length desirable for read on system bus when DMA is used. This is the maximum burst length, actual burst length may be less than this depending on other factors, such as block boundary  0000b - 16 transfers in a single burst 0001b - 1 transfers in a single burst 0010b - 2 transfers in a single burst 1111b - 15 transfers in a single burst
24 —	Reserved
25-31 RD_WML	Read watermark level. The number of words (32-bit) used as the watermark level (FIFO threshold) for SD read in CPU polling mode.  0000000b - 128 words 0000001b - 1 word 0000010b - 2 words 1111111b - 127 words

### 22.3.21 Force event register (FEVT)

#### 22.3.21.1 Offset

Register	Offset
FEVT	50h

### 22.3.21.2 Function

The FEVT is not a physically implemented register. Rather, it is an address at which the interrupt status register (IRQSTAT) can be written if the corresponding bit of the interrupt status enable register (IRQSTATEN) is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register actually sets the corresponding bit of interrupt status register (IRQSTAT). A read from this register always results in zeros.

### 22.3.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	Reserved				FEVTDMAE				FEVTDEB E				FEVTDC E				
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	Reserved								FEVTCIBAC12E	Reserved			FEVTAC12IE	FEVTAC12EBE	FEVTAC12CE	FEVTAC12TOE	FEVTAC12NE
W									0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 22.3.21.4 Fields

Field	Function
0-2	Reserved
—	
3 FEVTDMAE	Force event DMA error. Forces the IRQSTAT[DMAE] to be set.
4-5	Reserved
—	
6	Force event ADMA error. Forces the IRQSTAT[ADMAE] to be set.

Table continues on the next page...

## eSDHC register descriptions

Field	Function
FEVTADMAE	
7 FEVTAC12E	Force event Auto CMD12 error. Forces IRQSTAT[AC12E] to be set.
8 —	Reserved
9 FEVTDEBE	Force event data end bit error. Forces IRQSTAT[DEBE] to be set.
10 FEVTDCE	Force event data CRC error. Forces IRQSTAT[DCE] to be set.
11 FEVTDTOE	Force event data time out error. Forces IRQSTAT[DTOE] to be set.
12 FEVTCIE	Force event command index error. Forces the IRQSTAT[CCE] to be set.
13 FEVTCEBE	Force event command end bit error. Forces IRQSTAT[CEBE] to be set.
14 FEVTCCE	Force event command CRC error. Forces IRQSTAT[CCE] to be set.
15 FEVTCTOE	Force event command time out error. Forces IRQSTAT[CTOE] to be set.
16-23 —	Reserved
24 FEVTCNIBAC12E	Force event command not executed by Auto CMD12 error. Forces AUTOC12ERR[CNIBAC12E] to be set.
25-26 —	Reserved
27 FEVTAC12IE	Force event Auto CMD12 index error. Forces AUTOC12ERR[AC12IE] to be set.
28 FEVTAC12EBE	Force event Auto CMD12 end bit error. Forces AUTOC12ERR[AC12EBE] to be set.
29 FEVTAC12CE	Force event Auto CMD12 CRC error. Forces AUTOC12ERR[AC12CE] to be set.
30 FEVTAC12TOE	Force event Auto CMD12 time out error. Forces the AUTOC12ERR[AC12TOE] to be set.
31 FEVTAC12NE	Force event Auto CMD12 not executed. Forces AUTOC12ERR[AC12NE] to be set.

## 22.3.22 ADMA error status register (ADMAES)

### 22.3.22.1 Offset

Register	Offset
ADMAES	54h

### 22.3.22.2 Function

When an ADMA error interrupt occurs, the ADMA error states field in the ADMAES holds the ADMA state and the ADMA system address register (ADSADDR) holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST\_STOP: Previous location set in the ADMA system address register (ADSADDR) is the error descriptor address
- ST\_FDS: Current location set in the ADMA system address register (ADSADDR) is the error descriptor address
- ST\_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error
- ST\_TFR: Previous location set in the ADMA system address register (ADSADDR) is the error descriptor address

In case of a write operation, the host driver should use the ACMD22 to get the number of the written block rather than using this information, since unwritten data may exist in the host controller.

The Host controller generates the ADMA error interrupt when it detects invalid descriptor data (valid=0) in the ST\_FDS state. The host driver can distinguish this error by reading the valid bit of the error descriptor.

**Table 22-9. ADMA Error State Coding**

D30-D31	ADMA Error State (When Error Has Occurred)	Contents of ADMA System Address Register
00	IDLE (idle)	Current descriptor address on which ADMA error occurred
01	FETCH_DESC (fetch descriptor)	Current descriptor address on which ADMA error occurred
10	DATA_XFER (data transfer)	Current descriptor address on which ADMA error occurred
11	WAIT_STOP (Wait for ADMA to stop)	Current descriptor address on which ADMA error occurred

### 22.3.22.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R												ADMAIBE	ADMADCE	ADMALME	ADMAES	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.22.4 Fields

Field	Function
0-26 —	Reserved
27 ADMAIBE	ADMA internal bus error. This bit indicates that a system bus error occurred while ADMA transaction was underway. It is set when error response is received on the system bus. 0b - No error 1b - Error
28 ADMADCE	ADMA descriptor error. This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error
29 ADMALME	ADMA length mismatch error. This error occurs in the following two cases: <ul style="list-style-type: none"><li>While XFERTYP[BCEN] is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length</li><li>Total data length can not be divided by the block length</li></ul> 0b - No error 1b - Error
30-31 ADMAES	ADMA error state (when ADMA error is occurred). This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. Refer to the table above for more details.

## 22.3.23 ADMA system address register (ADSADDR)

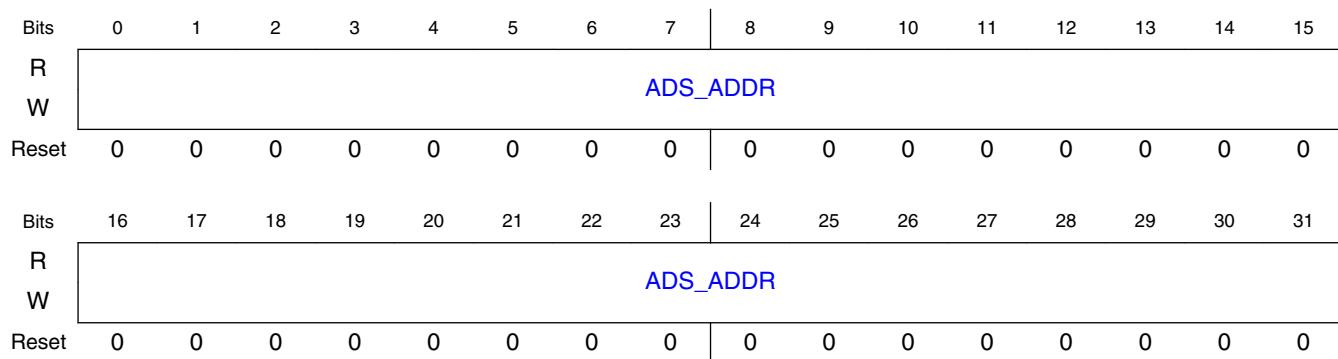
### 22.3.23.1 Offset

Register	Offset
ADSADDR	58h

### 22.3.23.2 Function

The ADSADDR contains the physical system memory address used for ADMA transfers.

### 22.3.23.3 Diagram



### 22.3.23.4 Fields

Field	Function
ADS_ADDR	<p>ADMA system address. This register holds 32-bit address of executing command of the descriptor table. At the start of ADMA, the host driver should set start address of the descriptor table. The ADMA increments this register address, which points to next line, when every fetching a descriptor line. When the ADMA error interrupt is generated, this register should hold valid descriptor address depending on the ADMA state. The host driver should program descriptor table on 32-bit boundary and set 32-bit boundary address to this register.</p> <p>It can be accessed only when no transaction is executing (that is, after a transaction has stopped). The host driver should initialize this register before starting an ADMA transaction.</p>

## 22.3.24 Host controller version register (HOSTVER)

### 22.3.24.1 Offset

Register	Offset
HOSTVER	FCh

### 22.3.24.2 Function

The HOSTVER contains the vendor host controller version information. All bits are read only and will read the same as the power-reset value.

### 22.3.24.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	VVN								SVN							
W																
Reset	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0

### 22.3.24.4 Fields

Field	Function
0-15 —	Reserved
16-23 VVN	<p>Vendor version number. These status bits are reserved for the vendor version number. The host driver should not use this status.</p> <p>Patterns not shown are reserved.</p> <ul style="list-style-type: none"> <li>00000000b - eSDHC Version 1.0</li> <li>00010000b - eSDHC Version 2.0</li> <li>00010001b - eSDHC Version 2.1</li> <li>00010010b - eSDHC Version 2.2</li> <li>00010011b - eSDHC Version 2.3</li> </ul>

Table continues on the next page...

Field	Function
	00100000b - eSDHC Version 3.0 00100001b - eSDHC Version 3.1 00100010b - eSDHC Version 3.2
24-31 SVN	Specification version number. These status bits indicate the host controller specification version. Patterns not shown are reserved. 00000000b - SD Host Specification Version 1.0 00000001b - SD Host Specification Version 2.0, supports test event register , and ADMA 00000010b - SD Host Specification Version 3.0

## 22.3.25 DMA error address register (DMAERRADDR)

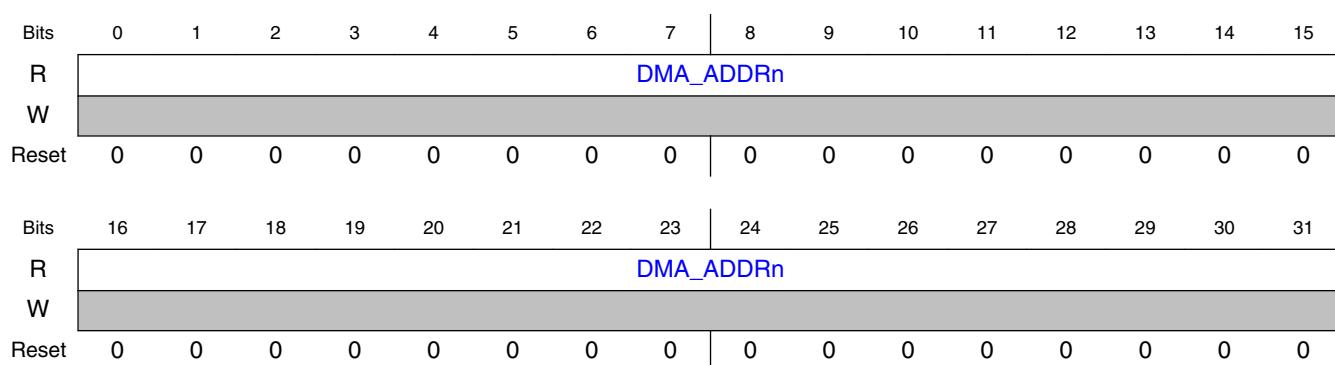
### 22.3.25.1 Offset

Register	Offset
DMAERRADDR	104h

### 22.3.25.2 Function

The DMAERRADDR contains the address of the transaction on which DMA error occurred.

### 22.3.25.3 Diagram



## 22.3.25.4 Fields

Field	Function
0-31 DMA_ADDRn	DMA error address. This field contains the system address of the transaction on which DMA error occurred.

## 22.3.26 DMA error attribute register (DMAERRATTR)

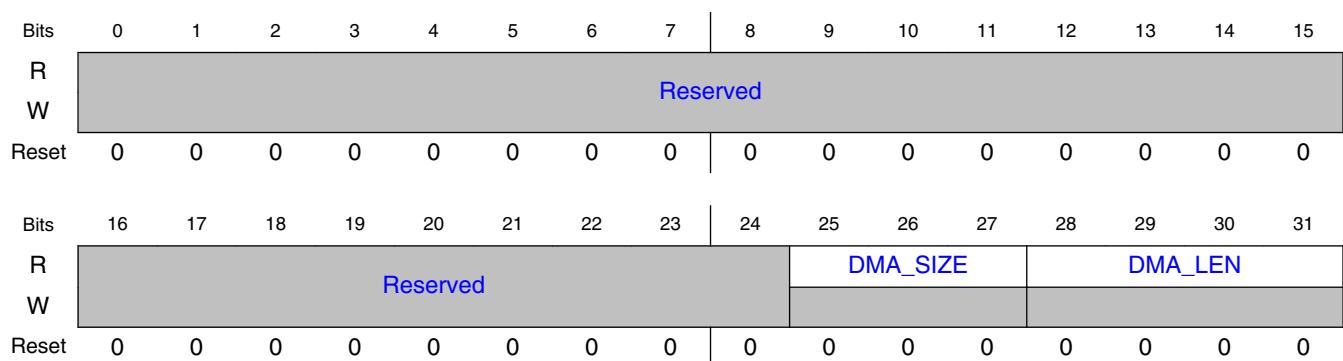
### 22.3.26.1 Offset

Register	Offset
DMAERRATTR	10Ch

### 22.3.26.2 Function

The DMAERRATTR contains attributes of the transaction on which DMA error occurred.

### 22.3.26.3 Diagram



### 22.3.26.4 Fields

Field	Function
0-24	Reserved

Table continues on the next page...

Field	Function
—	
25-27 DMA_SIZE	System bus burst size. This field contains burst size of the transaction on which DMA error occurred.
28-31 DMA_LEN	System bus burst length. This field contains burst length of the transaction on which DMA error occurred.

## 22.3.27 Host controller capabilities register 2 (HOSTCAPBLT2)

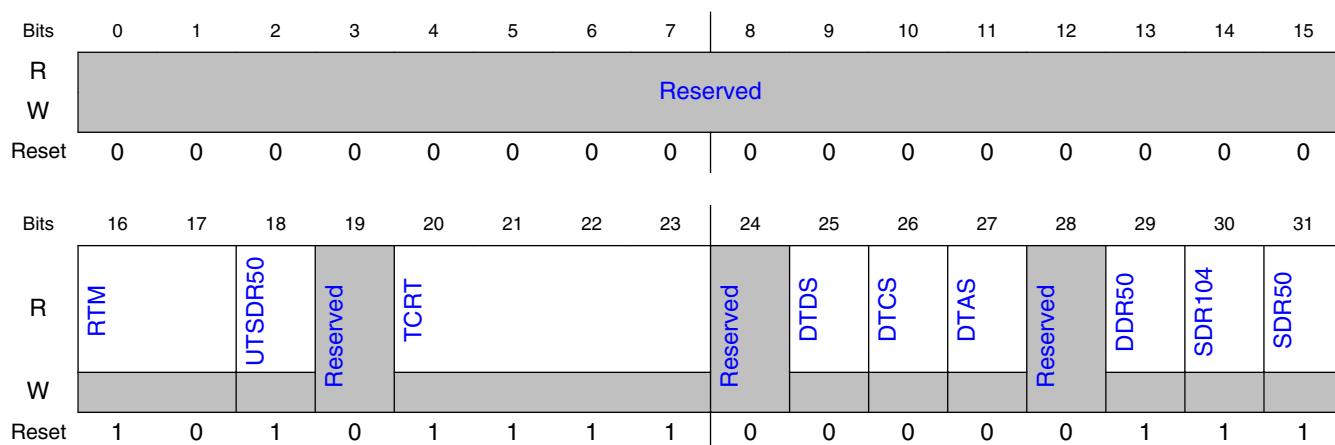
### 22.3.27.1 Offset

Register	Offset
HOSTCAPBLT2	114h

### 22.3.27.2 Function

This register provides the host driver with information specific to the eSDHC implementation. The value in this register is the power-on-reset value, and does not change with a software reset. Any write to this register is ignored.

### 22.3.27.3 Diagram



## 22.3.27.4 Fields

Field	Function
0-15 —	Reserved
16-17 RTM	Re-tuning modes. This bit indicates the supported re-tuning modes for UHS. 00b - Mode 1 - Software Timer 01b - Mode 2 - Software Timer and Re-tuning request 10b - Mode 3 - Software Timer, and Auto Re-tuning during data transfer 11b - Reserved
18 UTSDR50	Use tuning for SDR50. This bit indicates whether the host support tuning for SDR50 mode. 0b - Tuning for SDR50 mode not supported 1b - Tuning for SDR50 mode supported
19 —	Reserved
20-23 TCRT	Timer Count for Re-Tuning : This field indicates an initial value of Re-Tuning timer for retuning mode 1 to 3. 0000b - Re-Tuning timer disabled 0001b - 1 second 0010b - 2 seconds 0011b - 4 seconds 1011b - 1024 seconds 1100-1110b - Reserved 1111b - Get timer information from other source.
24 —	Reserved
25 DTDS	Driver type D support. This bit indicates whether the system is capable of using driver type D. 0b - Driver Type D not supported 1b - Driver Type D Supported
26 DTCS	Driver type C support. This bit indicates whether the system is capable of using driver type C. 0b - Driver Type C not supported 1b - Driver Type C Supported
27 DTAS	Driver type A support. This bit indicates whether the system is capable of using driver type A. 0b - Driver Type A not supported 1b - Driver Type A Supported
28 —	Reserved
29 DDR50	DDR50 support. This bit indicates whether the eSDHC supports the DDR mode. 0b - DDR mode not supported 1b - DDR mode supported
30 SDR104	SDR104 Support. This bit indicates whether the eSDHC supports the SDR104. 0b - SDR104 not supported 1b - SDR104 supported
31 SDR50	SDR50 support. This bit indicates whether the eSDHC supports the SDR50. 0b - SDR50 not supported 1b - SDR50 supported

## 22.3.28 Tuning block control register (TBCTL)

### 22.3.28.1 Offset

Register	Offset
TBCTL	120h

### 22.3.28.2 Function

This register contains fields for controlling the tuning block.

#### NOTE

Writing or reading to reserved fields of this register does not guarantee specific values will be written or read.

### 22.3.28.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	Reserved	Reserved	Reserved												
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R			Reserved		Reserved				Reserved	Reserved		Reserved	Reserved	TB_E_N	TB_MODE	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.28.4 Fields

Field	Function
0	Reserved

Table continues on the next page...

## eSDHC register descriptions

Field	Function
—	
1	Reserved
—	
2	Reserved
—	
3-7	Reserved
—	
8-15	Reserved
—	
16-17	Reserved
—	
18-19	Reserved
—	
20-23	Reserved
—	
24	Reserved
—	
25-26	Reserved
—	
27	Reserved
—	
28	Reserved
—	
29 TB_EN	Tuning block enabled. Tuning block should be enabled for high speed SDR mode more than 50 MHz SD clock frequency.  This bit is not reset by software reset for all.  0b - Tuning block is disabled 1b - Tuning block is enabled
30-31 TB_MODE	Tuning Mode  Tuning Mode. Selects tuning mode when tuning block is enabled. Refer to re-tuning modes in <a href="#">Table 22-10</a>  00b - Mode 1 - Software Timer 01b - Mode 2 - Software Timer, and Re-tuning request 10b - Mode 3 - Software Timer, and Auto Re-tuning during data transfer 11b - SW tuning mode - Software tuning mode where start and end point of data window needs to be programmed in TBPTR register and use software timer for re-tuning

### 22.3.29 Tuning block status register (TBSTAT)

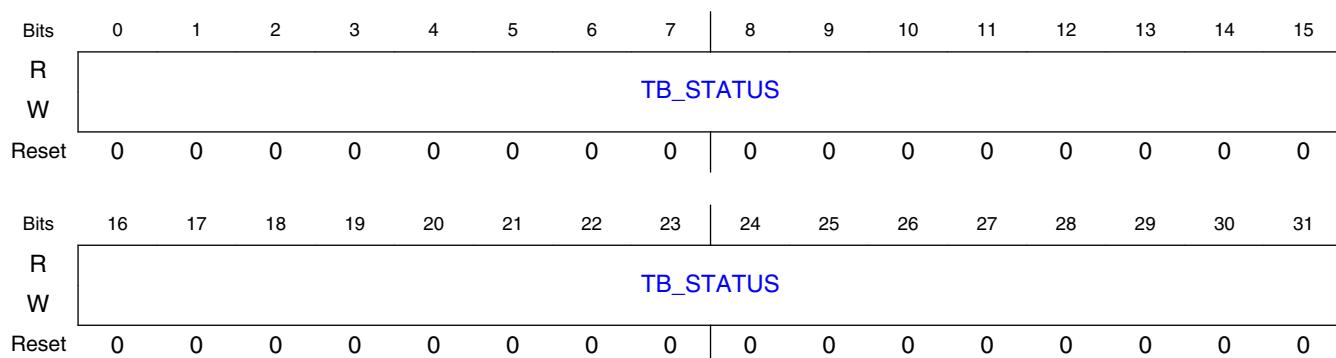
### 22.3.29.1 Offset

Register	Offset
TBSTAT	124h

### 22.3.29.2 Function

This register contains the status of the tuning block.

### 22.3.29.3 Diagram



### 22.3.29.4 Fields

Field	Function
0-31	Tuning Status.
TB_STATUS	

## 22.3.30 Tuning block pointer register (TBPTR)

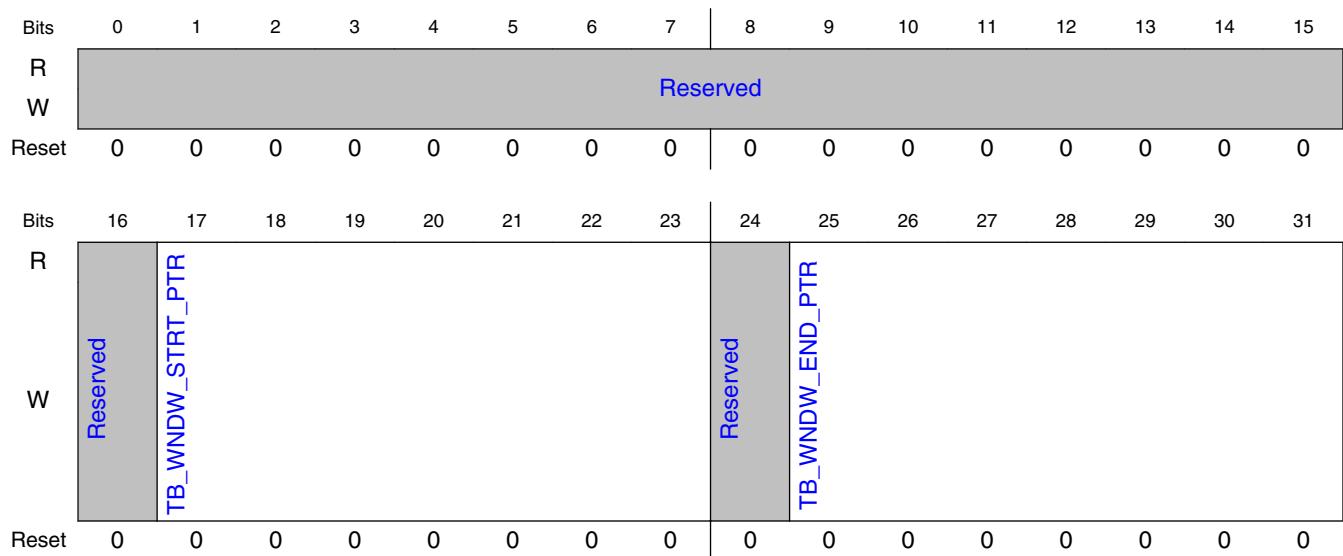
### 22.3.30.1 Offset

Register	Offset
TB PTR	128h

### 22.3.30.2 Function

This register contains fields for controlling tuning block pointers.

### 22.3.30.3 Diagram



### 22.3.30.4 Fields

Field	Function
0-16 —	Reserved
17-23 TB_WNDW_ST RT_PTR	Tuning window start pointer. Selects window start pointer for software tuning mode (when TBCTL[TB_MODE]=3).
24 —	Reserved
25-31 TB_WNDW_EN D_PTR	Tuning window end pointer. Selects window end pointer for software tuning mode (when TBCTL[TB_MODE]=3)

## 22.3.31 SD direction control register (SDDIRCTL)

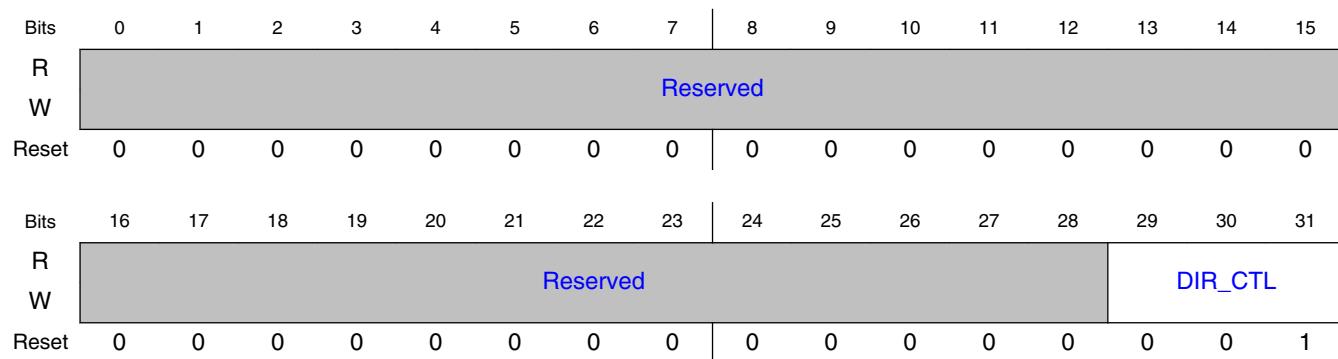
### 22.3.31.1 Offset

Register	Offset
SDDIRCTL	140h

### 22.3.31.2 Function

This register contains control for CMD and DAT lines direction.

### 22.3.31.3 Diagram



### 22.3.31.4 Fields

Field	Function
0-28 —	Reserved
29-31 DIR_CTL	Direction control Specify the turnaround time required for external transceiver after the assertion of direction pins in number of SD clocks 000b - No turnaround time required 001b - 1 SD clock period for turnaround 010b - 2 SD clock periods for turnaround 111b - 7 SD clock periods for turnaround

## 22.3.32 SD Clock Control Register (SDCLKCTL)

### 22.3.32.1 Offset

Register	Offset
SDCLKCTL	144h

### 22.3.32.2 Function

This register contains fields for controlling SD external and loopback clock.

### 22.3.32.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	LPBK_CLK_SEL	LPBK_SD_CLK_DL_Y_DIF	Reserved			LPBK_CLK_DL_Y										
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMD_CLK_CTL	Reserved		Reserved		Reserved										
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 22.3.32.4 Fields

Field	Function
0 LPBK_CLK_SE L	SD Loopback Clock Select. This field specifies whether SD clock is loopbacked from external pin or from internal pad.  0b - SD card clock is loopbacked from internal pad 1b - SD card clock is loopbacked from external pin
1 LPBK_SD_CLK _DLY_DIR	SD Loopback Clock Delay Direction. This field specifies whether SD loopback card clock is delayed in positive or negative direction.  0b - SD card loopback clock is delayed by LPBK_CLK_DLY value 1b - SD card loopback clock is early by LPBK_CLK_DLY value
2-3 —	Reserved
4-15 LPBK_CLK_DL Y	SD Loopback Clock Delay. This field specifies the number of peripheral clocks (based on ESDHCCTL[PCS]) by which SD loopback card clock is delayed.  000000000000b - No delay in SD card clock 000000000001b - 1/2 peripheral clocks delay introduced in SD card clock 000000000010b - 2/2 peripheral clocks delay introduced in SD card clock 111111111111b - 4095/2 peripheral clocks delay introduced in SD card clock
16 CMD_CLK_CTL	Command Logic Clock Control. This field specifies controls clock to the command logic.  0b - Command logic clock is same as data logic clock 1b - Command logic clock is 25% shifted early from data logic clock
17 —	Reserved
18-19 —	Reserved
20-31 —	Reserved

## 22.3.33 eSDHC control register (ESDHCTL)

### 22.3.33.1 Offset

Register	Offset
ESDHCTL	40Ch

### 22.3.33.2 Function

This register contains fields for controlling DMA transfers.

### 22.3.33.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								RTOCV		PCS		FAF		RTR	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								RD_DIS		SNOOP		Reserved		WR_BUF	
W	RD_PRFETCH_BLKCNT								0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 22.3.33.4 Fields

Field	Function
0-9 —	Reserved
10-11 RTOCV	Register timeout count value. This field define the timeout value for register access. 00b - Timeout count value for register access is $2^{10}$ clocks 01b - Timeout count value for register access is $2^{11}$ clocks 10b - Timeout count value for register access is $2^{12}$ clocks 11b - Timeout count value for register access is $2^{13}$ clocks
12 PCS	Peripheral clock select. This bit selects the clock used for generating SD clock. This bit is not reset by software reset for all. 0b - Platform clock is used 1b - Peripheral/2 clock is used
13 FAF	Flush asynchronous FIFO. This bit is used to flush asynchronous FIFO. It can be set by software and will be auto cleared by hardware. 0b - No Flush 1b - Flush async FIFO
14 RTR	Re-tuning request. This bit indicates re-tuning request status. eSDHC may request host driver to execute re-tuning sequence, by setting this bit, when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data. This bit is cleared when a command

Table continues on the next page...

Field	Function
	is issued with setting execute tuning in the system control 2 register. Changing of this bit from 0 to 1 generates re-tuning event. Refer to interrupt status registers for more detail. This bit isn't set to 1 if tuning block enable in the tuning control register is set to 0 (using fixed sampling clock). This is read-only field. 0b - No re-tuning request 1b - Re-tuning request is set
15 CRS	Clock register select. This bit selects the clock division format of SDCLKFS and DVS fields of system control register. 0b - SDCLKFS is defined as 8-bit field, and DVS is active in system control register 1b - SDCLKFS is defined as 10-bit field, CGS is active in system control register
16-18 —	Reserved
19-23 RD_PRFTCH_B LKCNT	Read prefetch block count. For DMA read (SD write), this field specifies the maximum number of SD blocks that the host can prefetch from the system memory. It can have following values: 00000b - No prefetch 00001b - 1 SD block prefetch 00010b - 2 SD block prefetch 11111b - 31 SD block prefetch
24 PAD_DIS	Pad disable Pad disable. Padding disable control for DMA transaction with non-word (4-bytes) aligned block size. For more details refer to <a href="#">Data buffer and block size</a> . 0b - DMA will pad data at the end each block transfer. 1b - DMA will not pad data at the end of each block transfer.
25 SNOOP	Snoop attribute. Snoop enable for DMA transaction. 0b - DMA transactions are not snooped by the CPU data cache. 1b - DMA transactions are snooped by the CPU data cache.
26-27 —	Reserved
28 WR_BUF	Write bufferable. DMA always initiate write transaction on System bus corresponding to last in every SD block as non-bufferable. This bit specifies whether all non-last write transactions will be bufferable or not. 0b - Non-last write transactions are not bufferable. 1b - Non-last write transactions are bufferable.
29 RD_SAFE	Read safe. This bit should be set only if the target of read dma operation is a well behaved memory which is not affected by the read operation and will return the same data if read again from the same location. This means that unaligned reading operation can be rounded up to enable more efficient read operations. 0b - It is not safe to read more bytes than were intended. 1b - It is safe to read more bytes than were intended.
30-31 —	Reserved

## 22.4 Functional description

The eSDHC block is partitioned in five major sub-blocks as shown in [Figure 22-2](#).

- SD interface and control unit

This block interfaces with the SD bus. It is mainly responsible for controlling the SD bus operation and transferring data from Data Buffer and SD bus. It also interacts with System Interface and Control Unit.

- System interface and control unit

This block interfaces with the system interfaces (that is, the system bus in DMA mode and register bus in CPU polling mode) and transfers the card data from the data buffer and system.

- Register bank

This block interfaces with register bus and contains all the registers. It controls the overall operation of eSDHC and also provides status through various registers.

- Clock and reset

This block generates divided clock for SD interface and provides appropriate reset to all the blocks.

- SD monitor

This block monitors the SD bus and provides status to register banks, such as card interrupt, write protect and card detect.

## 22.4.1 System interface and control unit (SysICU)

The SysICU block is further partitioned into three major sub-blocks:

- System control block

This sub-block receives data transfer request from the register bank, controls data transfer for whole transaction and generates control signals for DMA and buffer control operation on per block basis.

- Buffer control block

This sub-block handles buffer port register access in CPU polling mode and contains the buffer FIFO to store transfer data. It controls the data transfer per-block basis in the CPU polling mode.

- DMA block

This sub-block generates DMA transactions on system bus to transfer the data between system memory and data buffer.

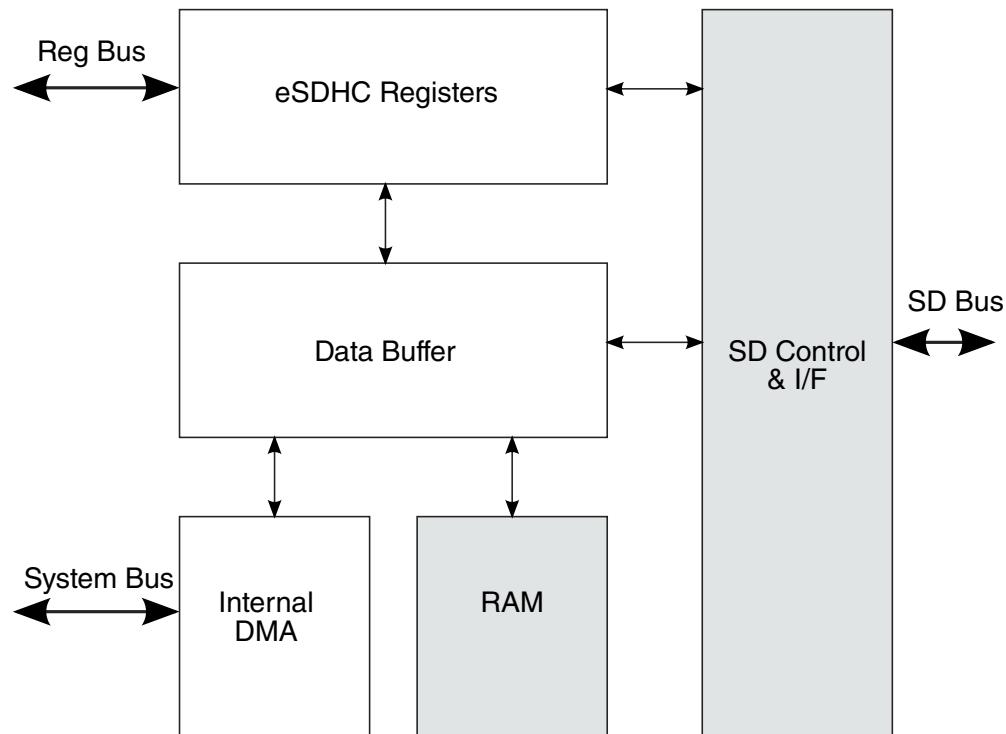
The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA system interface, register bank as well as IP bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

### 22.4.1.1 Data buffer

The eSDHC uses one configurable data buffer, so that data can be transferred between the system/register bus and SD card in an optimized manner to maximize throughput between the two clock domains. See the figure below for illustration of the buffer scheme.

The buffer is used as temporary storage for data being transferred between the host system and the card.

The watermark levels for read and write are both configurable, and can be any number from 1-128 words for CPU polling mode. The burst lengths for read and write are also configurable, and can be any number from 1-16 for DMA mode.



**Figure 22-3. eSDHC buffer scheme**

There are two transfer modes to access the data buffer:

- CPU polling mode

Data is transferred over register bus. For a host read operation, when the number of words received in the buffer meets or exceeds the RD\_WML watermark value, then by monitoring (polling or interrupt) the BRR bit the host driver can read the buffer data port register (DATPORT) to fetch the amount of words set in the RD\_WML register from the buffer. For block size integral multiple of watermark level value set in watermark level register (WML), driver must access buffer data port register (DATPORT) exactly the same number of times as the watermark level. However, if the block size is not the times of the watermark level value, the software must access exactly the remained number of words at the end of each block. For example, for read operation, if the RD\_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block count is 2, then the access times to Data Buffer Port Register for the burst sequence in the whole transfer process must be 4, 4, 2(for first block); 4, 4, 2(for second block). The write operation is similar.

- DMA mode (includes simple and advanced DMA accesses)

Data is transferred over system bus. DMA interrupt is generated after all the data is transferred to/from the buffer.

#### **22.4.1.1.1 Write operation sequence**

There are two ways to write data into the buffer when the user transfers data to the card:

- By processor core polling through IRQSTAT[BWR] (interrupt or polling)
- By using the DMA

When the DMA is not used, (CPU polling mode, that is, the XFERTYP[DMAEN] is not set when the command is sent), and when the amount of buffer space exceeds the value set in the WR\_WML register, PRSSTAT[BWR] is set. The buffer write ready interrupt is generated if it is enabled by software.

When DMA is used, the eSDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the eSDHC aborts the data transfer and abandons the current block. If the current data transfer is in multi block mode, the eSDHC does not automatically send CMD12, even though the XFERTYP[AC12EN] is set. The host driver needs to send CMD12 in this scenario. It is recommended that a Software Reset for Data be applied before the transfer is re-started after error recovery.

#### **22.4.1.1.2 Read operation sequence**

There are two ways to read data from the buffer when the user transfers data to the card:

- By processor core polling through IRQSTAT[BRR] (interrupt or polling)
- By using the DMA

When DMA is not used (CPU polling mode, that is, the XFERTYP[DMAEN] is not set when the command is sent), and when the amount of data exceeds the value set in the RD\_WML register, that is available and ready for system fetching data, PRSSTAT[BRR] is set. The buffer read ready interrupt will be generated if it is enabled by software.

When DMA is used, the eSDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the eSDHC will abort the data transfer and abandon the current block. If the current data transfer is in multi block mode, the eSDHC will not automatically send CMD12, even though the XFERTYP[AC12EN] is set. The host driver should send CMD12 in this scenario. It is recommended that a software reset for data be applied before the transfer is re-started after error recovery.

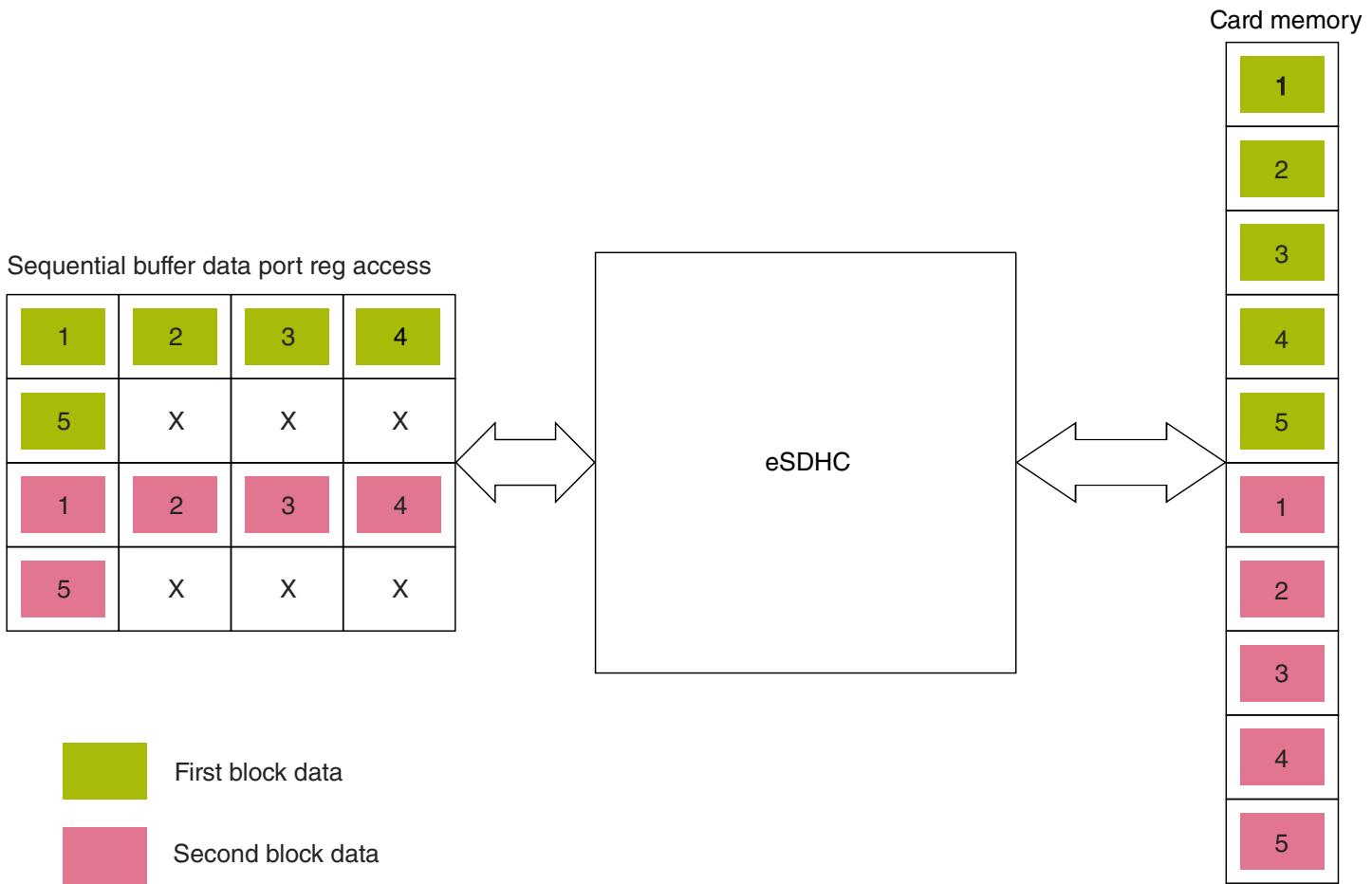
#### **22.4.1.1.3 Data buffer and block size**

In eSDHC, the data buffer can hold up to 128 words (32-bit).

The watermark levels for both write and read can be configured for CPU polling mode. The watermark level can be from one word to a maximum of 128 words. For both DMA read and write, the burst length can be configured from one to a maximum of 16. The host driver may configure watermark level and burst length value according to the system situation and requirement in the watermark level register (WML).

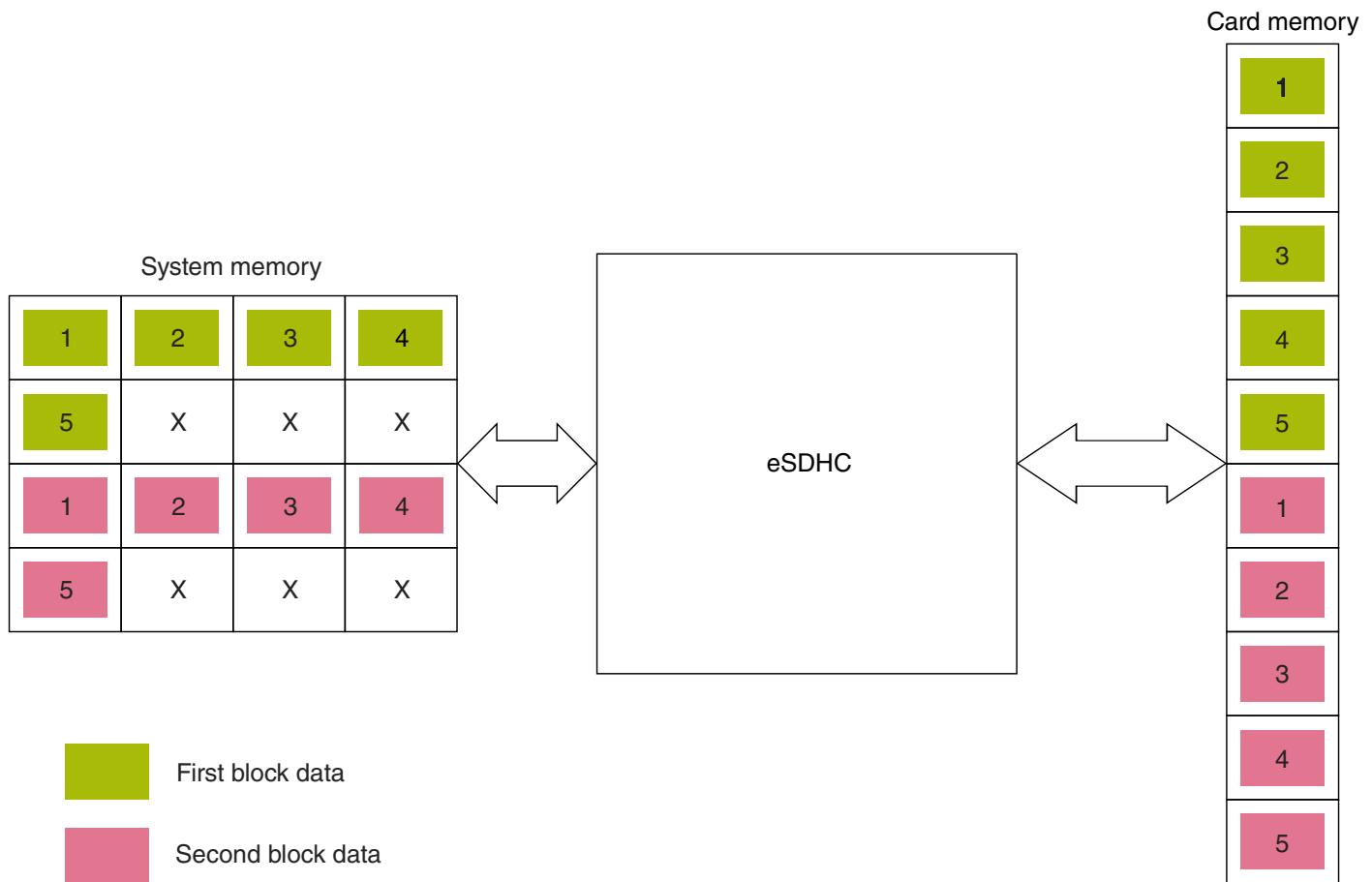
During a multi-block data transfer, the block length may be set to any value between 1 and 2048 bytes inclusive that satisfies the requirements of the external card. The only restriction to it can be from the external card, which may not support that large a block or partial access to block (which is not an integer times of 512 bytes).

For CPU polling mode, when block size not a multiple of four; that is, not word aligned, eSDHC requires stuff bytes at the end of each block, as defined in the host specification. For example, if the block size is 5 bytes and there are two blocks to write, there must be two register bus writes to buffer data port register (DATPORT) for each block; and for each block, the ending non-word aligned bytes should be stuffed in buffer data port register (DATPORT) write to make it word aligned. For this example, 3 bytes should be stuffed. eSDHC will transfer only the required number of bytes to the card and ignore the stuff bytes as shown in the figure below. Read operation is similar.

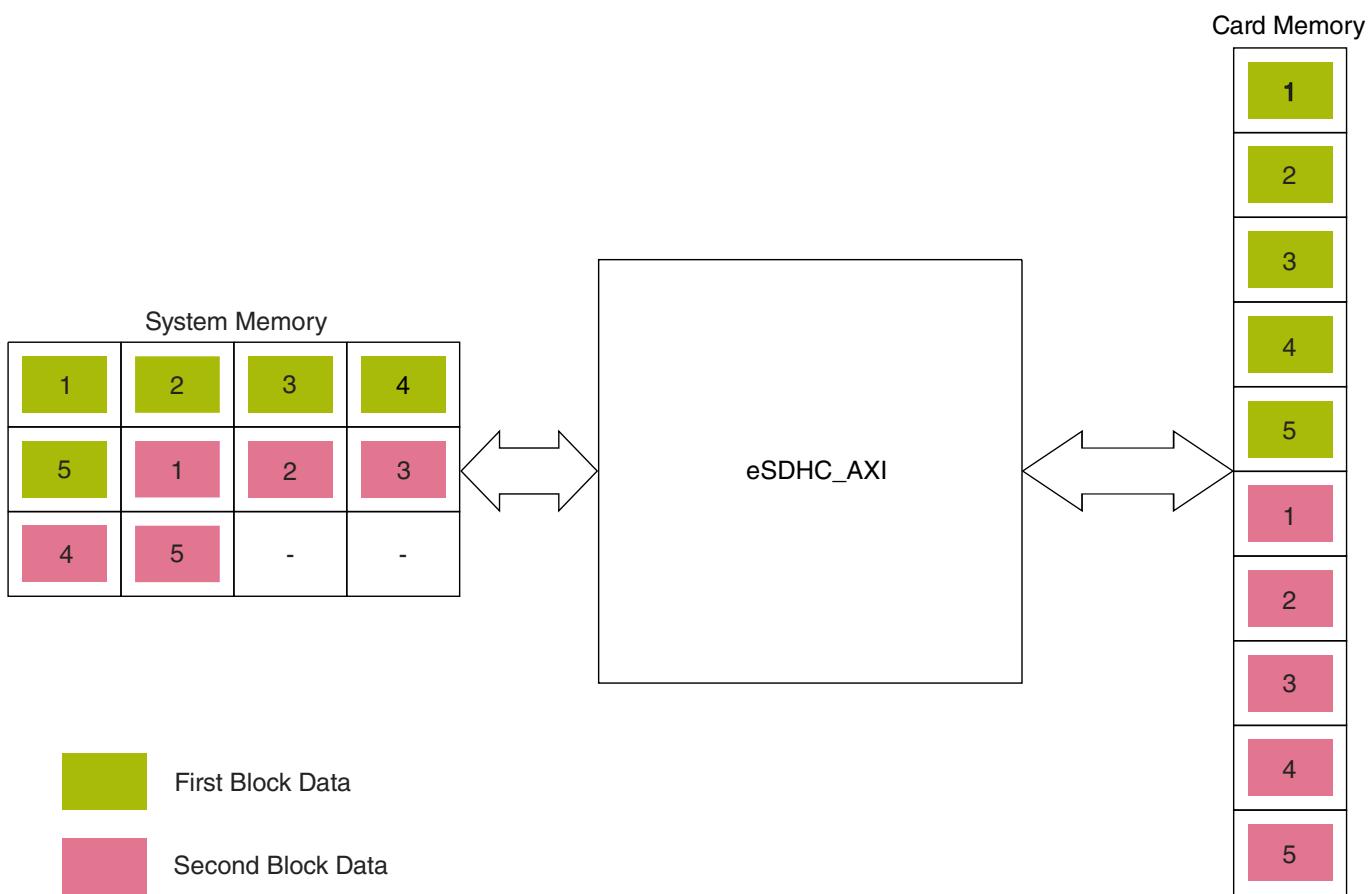


**Figure 22-4. Byte stuffing for CPU polling mode**

For DMA mode, when block size not a multiple of four; that is, not word aligned, eSDHC requires stuff bytes depending on the PAD\_DIS field programmed in DMA Control register. The data transfer with byte stuffing enabled (when DMACTL[PAD\_DIS]=0) is shown in [Figure 22-5](#). And, data transfer with byte stuffing disabled (when DMACTL[PAD\_DIS]=1) is shown in [Figure 22-6](#). Transfer with byte stuffing disabled eliminates the software overhead of byte stuffing. Driver may program DMA Control register accordingly.



**Figure 22-5. Byte stuffing for DMA mode when `DMACTL[PAD_DIS]=0`**



**Figure 22-6. No byte stuffing for DMA mode when DMACTL[PAD\_DIS]=1**

#### 22.4.1.1.4 Dividing large data transfer

This SDIO command CMD53 definition, limits the maximum data size of data transfers according to the following formula:

$$\text{Max data size} = \text{Block size} \times \text{Block count}$$

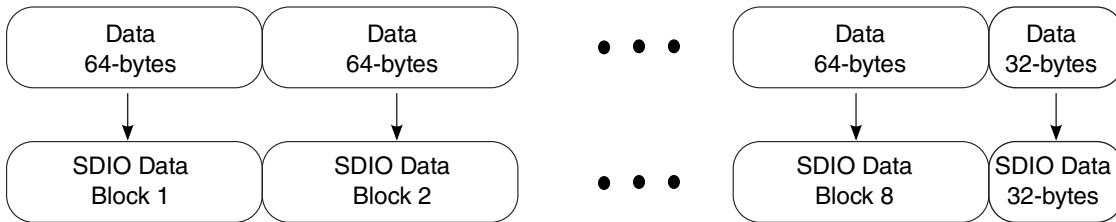
The length of a multiple block transfer needs to be in block size units. If the total data length cannot be divided evenly into a multiple of the block size, then based on the function and the card design, there are two ways to transfer the data. First option is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Second option is to add dummy data in the last block to fill the block size provided the card manages the removal of the dummy data.

Figure below shows an example which explains the dividing of large data transfers. In this figure, assume a kind of WLAN SDIO card that only supports block size up to 64 bytes. Although the eSDHC supports a block size of up to 2048 bytes, the SDIO can only accept a block size less than 64 bytes. Thus, the data must be divided (see example below).

544-bytes WLAN Frame



WLAN Frame is divided equally into 64-byte blocks plus the remainder 32-bytes



Eight 64-byte blocks are sent in Block Transfer Mode and the remainder 32-bytes are sent in Byte Transfer Mode

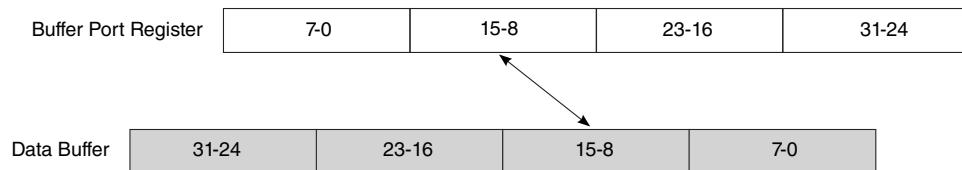
**Figure 22-7. Example for dividing large data transfers**

### 22.4.1.1.5 Byte order (endianness) of buffer data port register

For CPU polling mode, sequential and contiguous access is necessary to ensure the pointer address value is correct. Random or skipped access is not possible.

The byte order of buffer data port register (DATPORT), by reset is little endian mode. The data buffer is always little endian. The data is swapped inside the buffer, according to the endian mode configured by software in PROCTL[EMODE].

In little endian mode, the data is transferred between data buffer and buffer data port register (DATPORT) without any swapping. In big endian mode, the data between data buffer and buffer data port register (DATPORT) is byte-swapped as shown in the figure given below. For an SD write operation, byte order is swapped after data is fetched from the buffer port register and sent to data buffer. For a host read operation, byte order is swapped after the data is read from data buffer and sent to buffer port register.

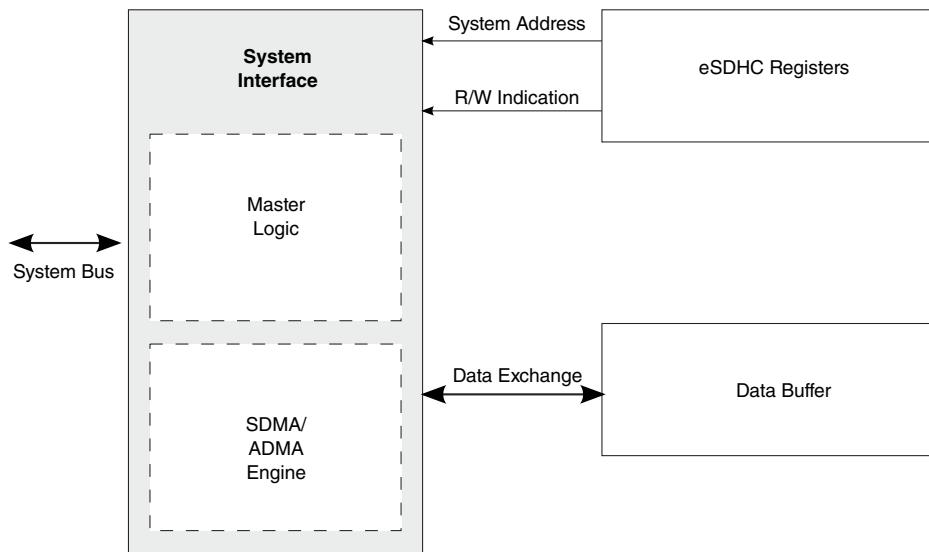


**Figure 22-8. Data swap between buffer data port register and data buffer in big endian mode**

### 22.4.1.2 DMA system interface

The DMA implements a DMA engine and the system master.

The eSDHC supports both SDMA and ADMA (ADMA1 and ADMA2). Figure below illustrates the DMA system interface block.



**Figure 22-9. DMA system interface block**

#### 22.4.1.2.1 DMA burst length

The actual burst length on system bus depends on the shortest of following factors:

- Burst length configured in the burst length field of the watermark level register (WML)
- Block size boundary
- Data boundary configured in the current descriptor (if the ADMA is active)

#### 22.4.1.2.2 System master interface

The System DMA engine can at times fail during data transfer.

When this error occurs:

- The DMA engine stops the transfer and goes to the error state.
- The internal data buffer stops accepting incoming data.
- The IRQSTAT[DMAE] is set to inform the driver.

Once the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DMAERRADDR[DMA\_ADDR] bits to get the starting address of the corrupted block. For error recovery, the software issues a data reset and re-starts the transfer from this address to recover the corrupted block.

### **22.4.1.3 Single DMA (SDMA)**

SDMA is single operation DMA, that is, data is transferred for single operation only (unlike ADMA which has chained descriptor table) from the starting address programmed in SDMA system address register (DSADDR) for entire data transfer size (block count x block size) as programmed in the block attributes register (BLKATTR) if XFERTYP[BCEN] is set.

#### **22.4.1.3.1 SDMA error**

SDMA will stop whenever an error is encountered on the system bus.

DMA error address register latches the transaction address on which the error occurred.

### **22.4.1.4 Advanced DMA (ADMA)**

Advanced DMA (ADMA) is a new DMA transfer algorithm, which is defined in the SD Host Controller Standard.

For single DMA, the data can be transferred for single operation only. The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. Higher speed DMA transfers are realized because the host MCU intervention is not needed during long DMA based data transfers.

The host controller has two types of ADMA: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory. ADMA2 improves the restriction so that data of any location and any size can be transferred in system memory. Their formats of descriptor table are different.

ADMA can recognize all kinds of descriptors defined in SD Host Controller Standard and if 'End' flag is detected in the descriptor, ADMA stops after this descriptor is processed.

eSDHC supports ADMA1 and ADMA2 with 32-bit addressing.

#### 22.4.1.4.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and End flag in descriptor

ADMA2 includes the following descriptors:

- Valid/Invalid descriptor
- Nop descriptor
- Rsv descriptor
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and End flag in descriptor

[Figure 22-10](#) explains the ADMA1 descriptor table format.

[Figure 22-12](#) explains the ADMA2 descriptor table format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it will ignore the high 32-bit. Address field should be set on word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

ADMA starts read/write operation after it reaches the Tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before Tran type descriptor. Every Tran type triggers a transfer, and the transfer data length is extracted from the most recent Set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 4 Kbyte if no set descriptor is ever met.

For ADMA2, Tran type descriptor contains both data length and transfer data address. Thus, only a Tran type descriptor can start a data transfer.

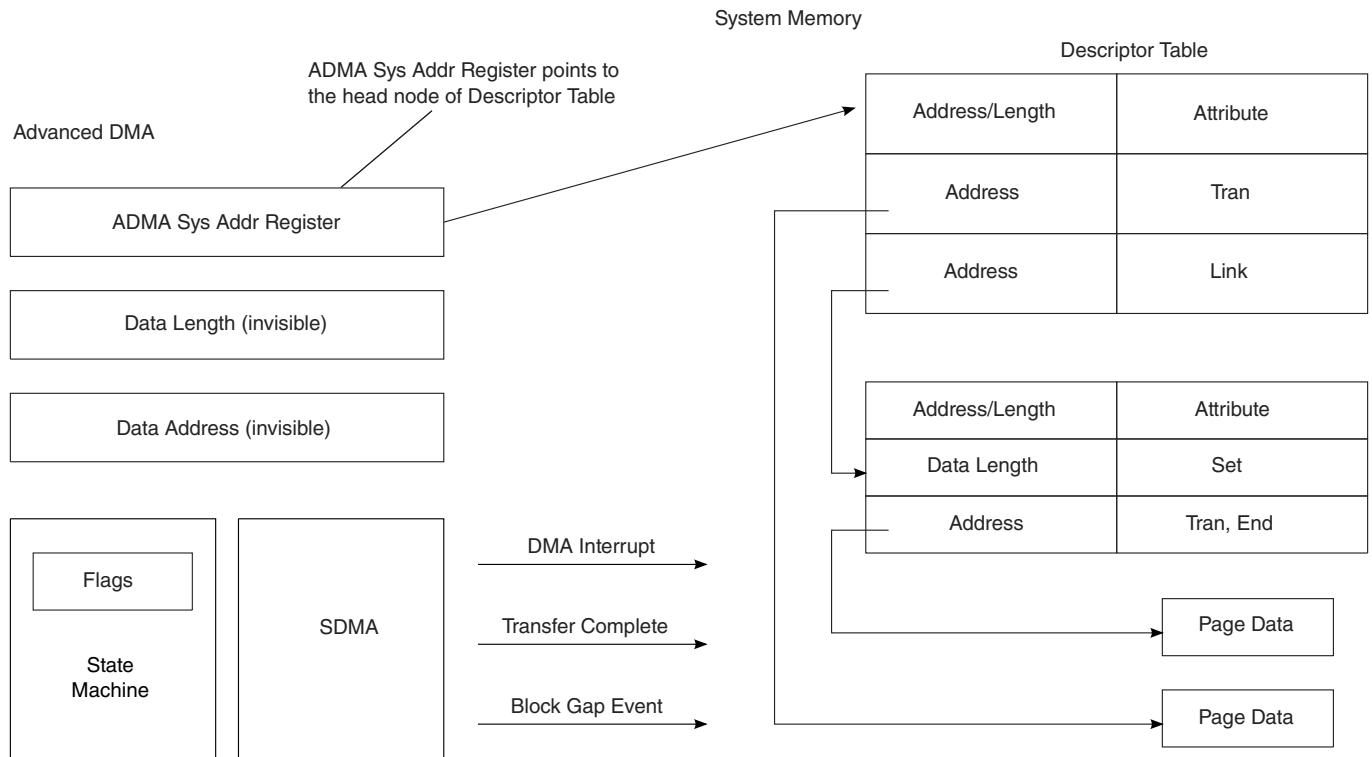
## Functional description

Address/Page Field		Address/Page Field		Attribute Field							
31	12	11	6	5	4	3	2	1	0		
Addressor Data Length		000000				Act2	Act1	0	Int	End	Valid

Act2	Act1	Symbol	Comment	31-28	27-12
0	0	NOP	No Operation	Do not care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid=1 indicates this line of descriptor is effective. If Valid=0, generate ADMA error interrupt and stop ADMA.
End	End=1 indicates current descriptor is the ending descriptor.
Int	Int=1 generates DMA Interrupt when this descriptor is processed.

**Figure 22-10. Format of the 32-bit address ADMA1 descriptor table**



**Figure 22-11. Concept and access method of ADMA1 descriptor table**

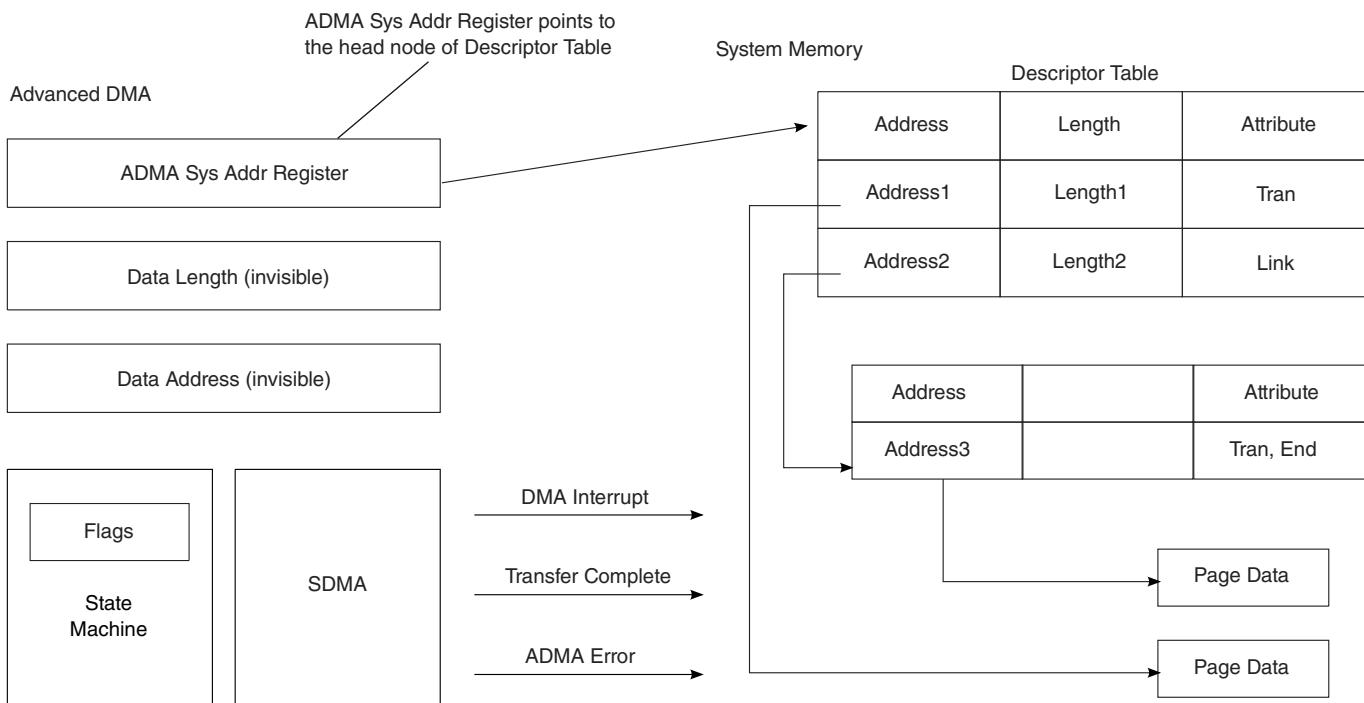
## Functional description

Address Field		Length		Reserved		Attribute Field						
63	32	31	16	15	6	5	4	3	2	1	0	
32-bit Address		16-bit Length		0000000000			Act2	Act1	0	Int	End	Valid

Act2	Act1	Symbol	Comment	Operation
0	0	NOP	No Operation	Do not care
0	1	Rsv	Reserved	Same as NOP. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid=1 indicates this line of descriptor is effective. If Valid=0, generate ADMA error interrupt and stop ADMA.
End	End=1 indicates current descriptor is the ending descriptor.
Int	Int=1 generates DMA Interrupt when this descriptor is processed.

**Figure 22-12. Format of the 32-bit address ADMA2 descriptor table**



**Figure 22-13. Concept and access method of ADMA2 descriptor table**

#### 22.4.1.4.2 ADMA interrupt

If the 'interrupt' flag of descriptor is set, ADMA will generate an interrupt, IRQSTAT[DINT] whenever the data transfer for the particular descriptor line is completed.

#### 22.4.1.4.3 ADMA error

The ADMA stops whenever any of the following error is encountered:

- Fetching descriptor error
- System response error
- Data length mismatch error

ADMA descriptor error is generated when it fails to detect 'Valid' flag in the descriptor. If ADMA descriptor error occurs, the interrupt is not generated even if the 'Interrupt' flag of this descriptor is set.

When the BLKCNTEN bit is set, the data transfer length set in buffer must be equal to the whole data transfer length set in the descriptor nodes, otherwise data length mismatch error is generated.

If the BLKCNTEN bit is not set, the whole data transfer length set in descriptor should be times of block length, otherwise, when all data set in the descriptor nodes are done, the data length mismatch error will occur.

The DMA error address register (DMAERRADDR) latches the transaction address on which the error occurred. The ADMA system address register (ADSADDR) latches the current descriptor line address which encountered the error.

### 22.4.2 SD interface and control unit (SDICU)

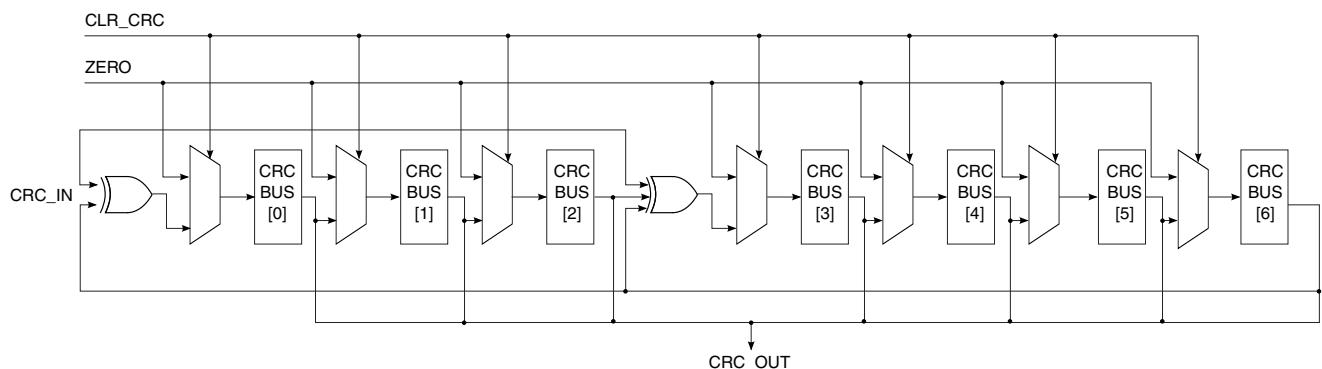
This block is partitioned into six major sub-blocks as shown in [Figure 22-2](#).

- Transfer control block: This sub-block receives command request from the register bank and overall controls other SDICU sub-blocks for SD bus operation on transaction level.
- SD command block: This sub-block controls the SD CMD line for sending command out. It receives the command request from Transfer Control block and sends it on SD CMD line.
- SD response block: This sub-block monitors the SD CMD line for receiving response and detecting line errors. It sends the command and response status to transfer control for transaction operation.

- SD data out block: This sub-block controls the SD DAT lines for sending out the block write data from the SD data out FIFO.
- SD data in block: This sub-block monitors the SD DAT line for detecting read and write data block end and busy period end for transfer control module. It also detects data line error and card interrupt.
- Tuning block: This sub-block receives Tuning block data from card and tunes IP. It then forwards the sampled data to Async FIFO for further operation.

### 22.4.2.1 Command CRC

See the figure below for an illustration of the structure for the command CRC shift register.



**Figure 22-14. Command CRC shift register**

The CRC polynomials for the CMD are as follows:

$$\begin{aligned}
 \text{Generator polynomial: } G(x) &= x^7 + x^3 + 1 \\
 M(x) &= (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + \dots + (\text{last bit}) * x^0 \\
 \text{CRC}[6:0] &= \text{Remainder } [(M(x) * x^7) / G(x)]
 \end{aligned}$$

### 22.4.2.2 Data CRC

The CRC polynomials for the DAT are as follows:

$$\begin{aligned}
 \text{Generator polynomial: } G(x) &= x^{16} + x^{12} + x^5 + 1 \\
 M(x) &= (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + \dots + (\text{last bit}) * x^0 \\
 \text{CRC}[15:0] &= \text{Remainder } [(M(x) * x^{16}) / G(x)]
 \end{aligned}$$

### 22.4.2.3 Tuning block (SDICU)

Tuning block is used for SD UHS SDR50, SDR104 and MMC HS200 modes.

Tuning block tunes the IP on tuning command (CMD19 for SD, CMD21 for MMC) data sent by card and the sampled data is sent to async FIFO for further operation.

This block oversamples data from card and selects particular sampling point after completion of tuning procedure, refer to [Tuning block procedure](#) for tuning block usage. Various re-tuning modes are supported by eSDHC as described below.

**Table 22-10. Re-tuning modes**

Re-Tuning Mode	Re-Tuning Method	Data Length
1	Software timer	4MB (Max.)
2	Software timer, and re-tuning request	4MB (Max.)
3	Software timer, and auto re-tuning during data transfer	Any

There are two re-tuning timings: Re-tuning request controlled by eSDHC and expiration of a re-tuning timer(software timer) controlled by the host driver. By receiving either timing, the host driver executes the re-tuning procedure just before a next command issue.

The maximum data length per read/write command is restricted so that re-tuning procedures can be inserted during data transfers.

### Re-tuning mode 1

eSDHC do not generate re-tuning request. In this case, the host driver should maintain all re-tuning timings by using a re-tuning timer. To enable inserting the re-tuning procedure during data transfers, the data length per read/write command shall be limited up to 4 MB.

### Re-tuning mode 2

eSDHC indicates the re-tuning timing by re-tuning request during data transfers. Then the data length per read/write command shall be limited up to 4 MB. During non data transfer, re-tuning timing is determined by re-tuning timer.

### Re-tuning mode 3

eSDHC takes care of the re-tuning during data transfer (auto re-tuning). Re-tuning request will not be generated and there is no limitation to data length per read/write command. During non data transfer, re-tuning timing is determined by re-tuning timer.

### Re-tuning timer control example for re-tuning mode 1

The software timer starts counting by loading the initial value. When the timer expires, the host driver marks an expiration flag. On receiving a command request, the host driver checks the expiration flag. If the expiration flag is set, then the host driver should perform

## Functional description

the re-tuning procedure before issuing a command. If the expiration flag is not set, then the host driver issues a command without performing the re-tuning procedure. Every time the re-tuning procedure is performed, the timer loads the new initial value and the expiration flag is cleared.

### Re-tuning timer control example for re-tuning mode 2 and mode 3

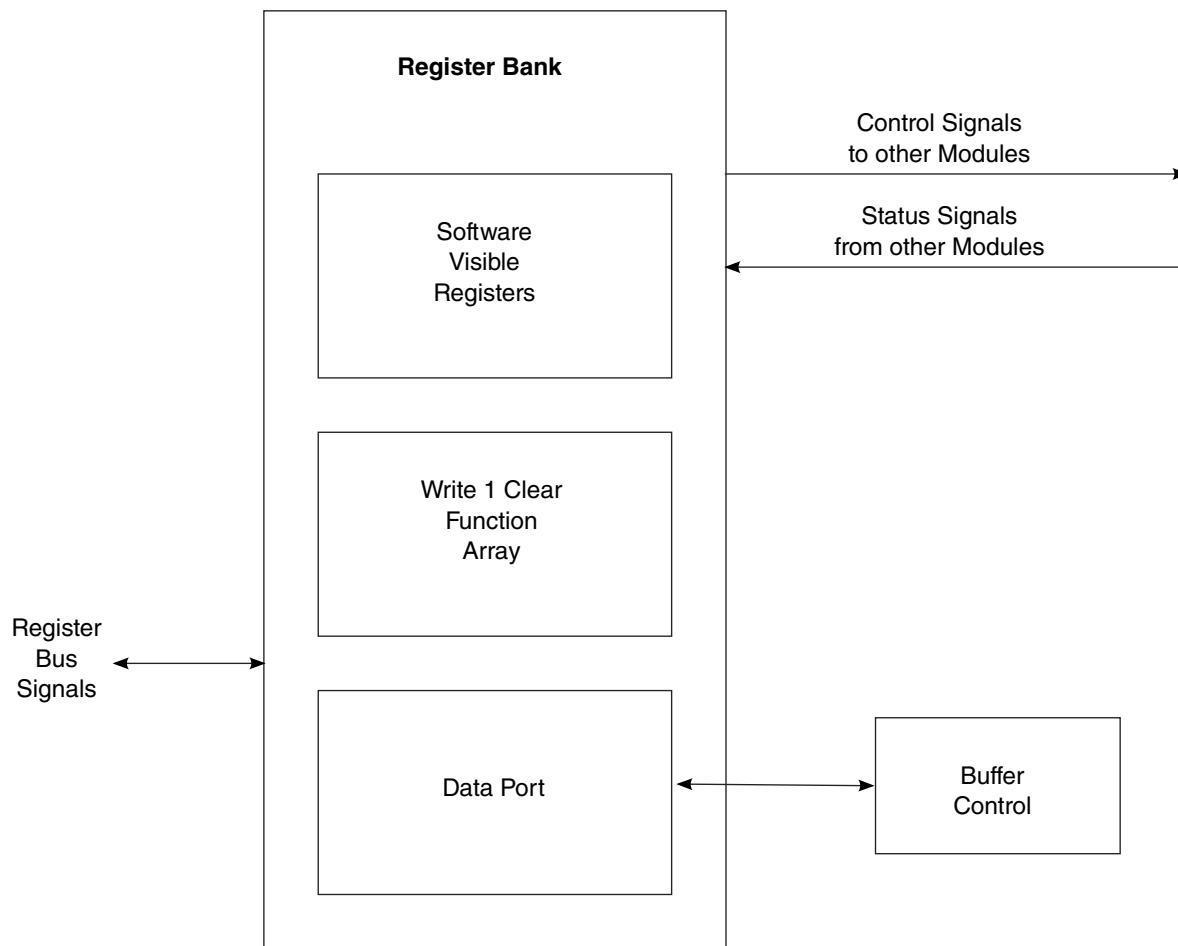
The software timer control is almost the same as re-tuning mode 1 except the timer loads the new initial value after data transfer (when receiving transfer complete). in case of mode 3, timer count for re-tuning is set either smaller value: tuning effective time after re-tuning procedure or after data transfer.

If a host system goes into power down mode, the host driver should stop the re-tuning timer and set the expiration flag to 1 when the host system resumes from power down mode.

### 22.4.3 Register bank

This block interfaces with register bus and it contains all the registers. It controls the overall operation of eSDHC and also provides status through various registers.

Register accesses is actually on the register bank. See the figure below for the block diagram.



**Figure 22-15. Register bank diagram**

Partial access is not allowed on any register; that is, it should be 32-bit access only.

#### 22.4.4 Clock and reset module

Clock and reset module generates divided clock for SD interface and provides appropriate reset to other modules.

There are four kinds of reset signals within eSDHC:

- Hardware reset
- Software reset for all
- Software reset for the data part
- Software reset for the command part

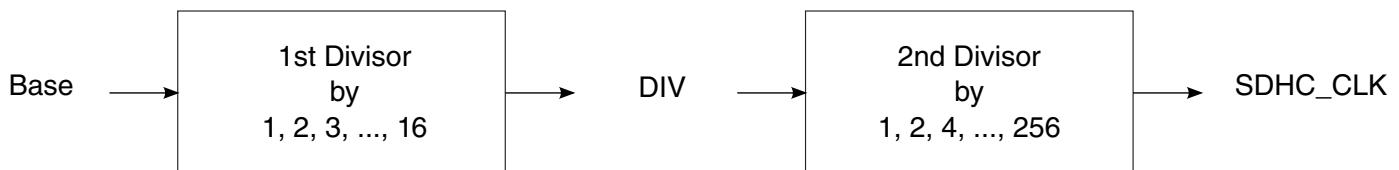
All these signals are fed into this module and stable signals are generated inside the module to reset all other modules.

If the internal data buffer is in danger, and the SD clock must be gated off to avoid buffer over/under-run, this module asserts the gate of the output SD clock to shut the clock off. After the buffer danger has recovered, and when the system access of the buffer catches up, the clock gate of this module opens and the SD clock becomes active again.

#### 22.4.4.1 Clock generator

The clock generator generates the SDHC\_CLK by source clock in two stages.

The figure below illustrates the structure of the divider. The term "Base" represents the frequency of the source clock.

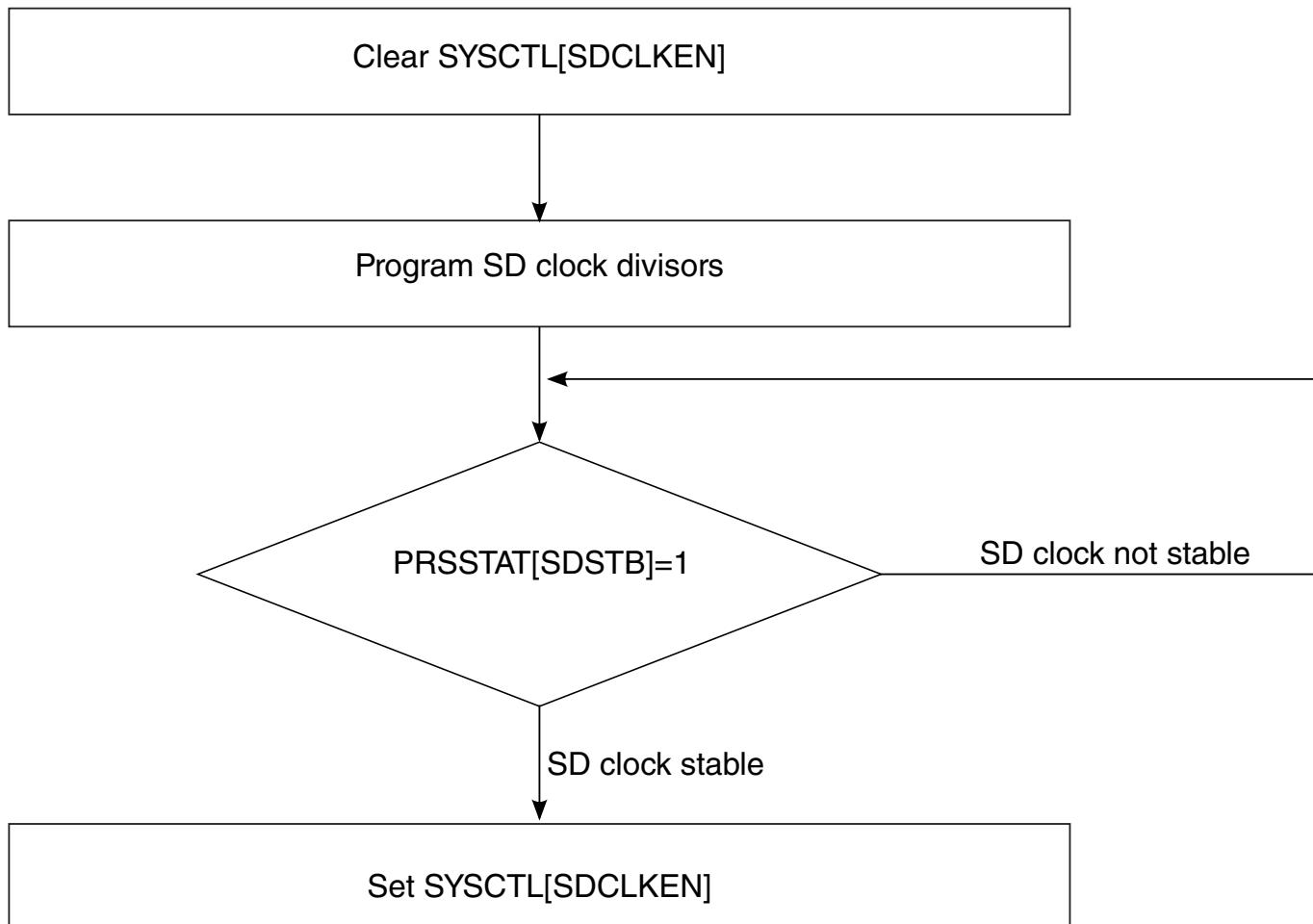


**Figure 22-16. Two stages of the clock divider**

The first stage outputs an intermediate clock (DIV), which can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler, and outputs the actual clock (SDHC\_CLK). This clock is the driving clock for sub modules SD Command and SD Data Out in SD Interface and Control Unit (refer to [Figure 22-2](#)) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage, can be DIV, DIV/2, DIV/4,..., or DIV/256. Thus, the highest frequency of the SDHC\_CLK is Base, and the next highest is Base/2, while the lowest frequency is Base/4096.

The figure below illustrates the sequence for changing the SD clock frequency.



**Figure 22-17. SD clock frequency change sequence**

Base clock can be selected by programming ESDHCCTL[PCS]. It selects between platform clock and peripheral clock / 2. Base clock is divided by two of peripheral clocks when PCS=1.

1. Clear SYSCTL[SDCLKEN]
2. Wait for PRSSTAT[SDSTB] to be set
3. Program appropriate value of ESDHCCTL[PCS]
4. Set SYSCTL[SDCLKEN]
5. Wait for PRSSTAT[SDSTB] to be set

## 22.4.5 SD monitor

The module detects the CD\_B (card detection) as well as the DAT3 signal. The transceiver reports the card insertion state according to the CD\_B state, the signal level on the DAT3 signal.

**NOTE**

Do not use DAT3 pin as a CD pin.

The module detects the WP (write protect) signal. With the information of the WP state, the register bank ignores the command, accompanied by a write operation, when the WP switch is on.

### 22.4.5.1 SDIO card interrupt

#### 22.4.5.1.1 Interrupts in 1-bit mode

In this case the DAT[1] pin is dedicated to providing the interrupt function.

An interrupt is asserted by pulling the DAT[1] low from the SDIO card, until the interrupt service is finished to clear the interrupt.

#### 22.4.5.1.2 Interrupt in 4-bit mode

Since the interrupt and DAT[1] share pin 8 in 4-bit mode, an interrupt is sent by the card and recognized by the host only during a specific time.

This is known as the interrupt period. eSDHC only samples the level on pin 8 during the interrupt period. At all other times, the host ignores the level on pin 8, and treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in 4-bit mode, there is only a limited period of time that the Interrupt Period can be active due to the limited period of data line availability between the multiple blocks of data. This requires a more strict definition of the Interrupt Period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DAT[1] line is held low for one clock cycle with the last clock cycle pulling DAT[1] high. On completion of the interrupt period, the card releases the DAT[1] line into the high Z state. eSDHC samples DAT[1] during the interrupt period when the PROCTL[IABG] is set.

Refer to SDIO card specification v2.0 for further information about the SDIO card interrupt.

### 22.4.5.1.3 Card interrupt handling

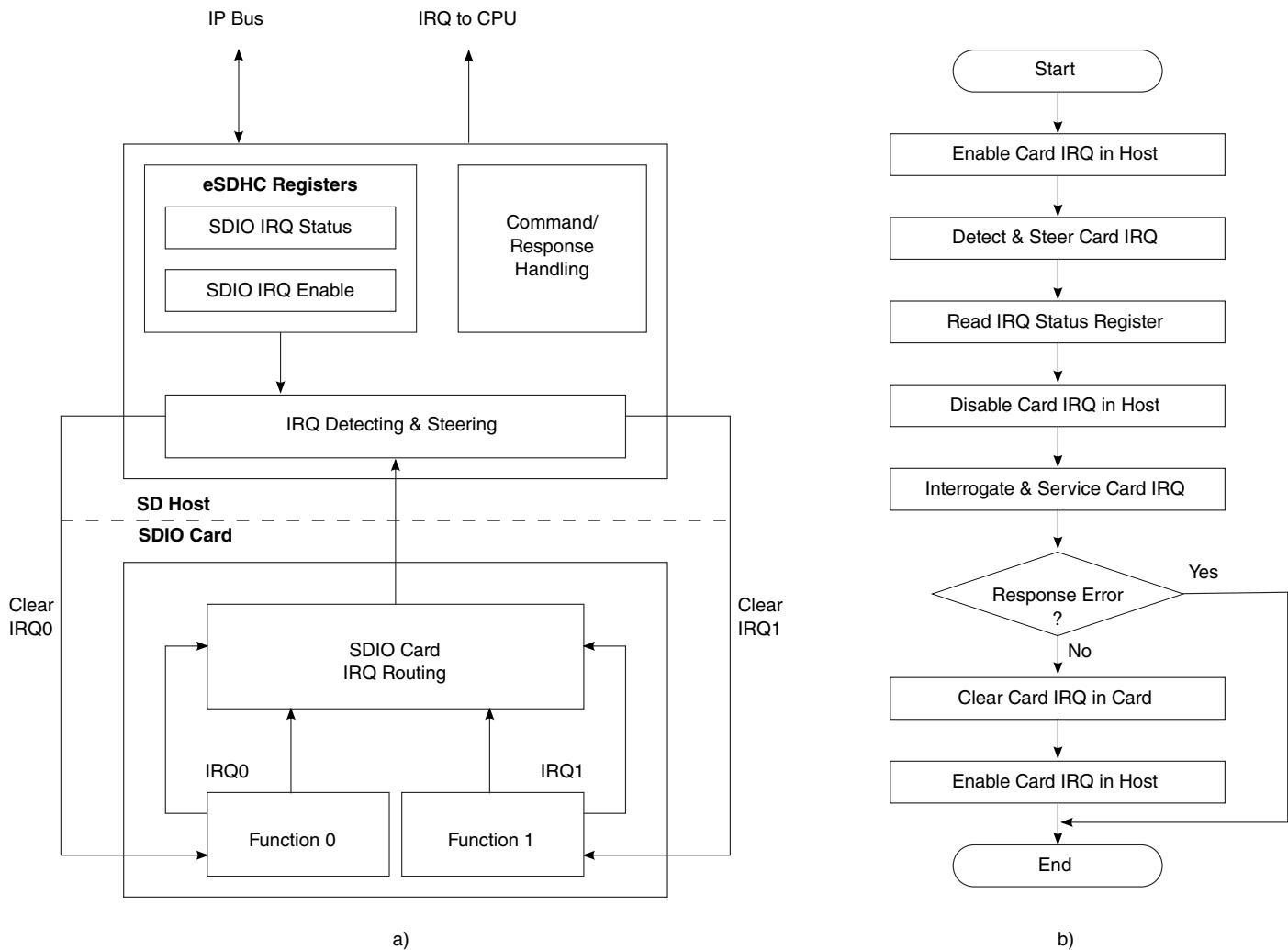
When the CINTIEN bit in the interrupt signal enable register is set to 0, the eSDHC clears the interrupt request to the host system.

The host driver should clear this bit before servicing the SDIO interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

In 1-bit mode, eSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the host system interrupt controller. When the SDIO status is set, and the host driver needs to service this interrupt, so the SDIO bit in the interrupt control register of SDIO card is cleared. This is required to clear the SDIO interrupt status latched in the eSDHC and to stop driving the interrupt signal to the system interrupt controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO interrupt enable bit is set to 1, and the eSDHC starts sampling the interrupt signal again.

The figure below illustrates the SDIO card interrupt scheme and sequences of software and hardware events that take place during a card interrupt handling procedure.

## Functional description



**Figure 22-18. Card interrupt scheme and card interrupt detection and handling procedure**

### 22.4.5.2 Card insertion and removal detection

When the DAT[3] pin is not used for card detection (for example, it is implemented in GPIO), the CD pin must be connected for card detection. Whether DAT[3] is configured for card detection or not, the CD pin is always a reference for card detection. Whether the DAT[3] pin or the CD pin is used to detect card insertion, the eSDHC sends an interrupt (if enabled) to inform the host system that a card is inserted.

### 22.4.5.3 Power management and wake up events

If no operation is expected to happen between eSDHC and the card through the SD bus, the user can completely disable the IP in the chip-level clock control module to save power.

When the user needs to use the eSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to the eSDHC are disabled, for instance, when the system is in low power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The eSDHC can generate these interrupt even when there are no clocks enabled. The three interrupts which can be used as wake up events are:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from SDIO card

The eSDHC offers a power management feature. By clearing the clock enabled bits in the system control register (SYSCTL), the clocks are gated in the low position to the eSDHC. For maximum power saving, the user can disable all the clocks to the eSDHC when there is no operation in progress.

These three wake up events (or wakeup interrupts) can also be used to wake up the system from low-power modes.

#### **NOTE**

To make the interrupt a wakeup event, when all the clocks to the eSDHC are disabled or when the whole system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to [Protocol control register \(PROCTL\)](#) for more information.

#### 22.4.5.3.1 Setting wake up events

For eSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters sleep mode.

Before the software disables the host clock, it should ensure that all of the following conditions are met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- Internal data buffer is empty

## 22.5 Initialization/application of eSDHC

All communication between the system and the cards are controlled by the host.

The host sends commands of two types: Broadcast and Addressed (point-to-point).

Broadcast commands are intended for all cards, such as "GO\_IDLE\_STATE", "SEND\_OP\_COND", "ALL\_SEND\_CID" and so on. In Broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to [Commands for MMC/SD/SDIO and](#) for the commands of bc and bcr categories.

After the Broadcast command CMD3 is issued, the cards enter standby mode. Addressed type commands are used from this point. In this mode, the CMD/DAT I/O pads will turn to push-pull mode, to have the driving capability for maximum frequency operation.

Refer to [Commands for MMC/SD/SDIO and](#), for the commands of ac and adtc categories.

### 22.5.1 Command send and response receive basic operation

Assuming the data type WORD is an unsigned 32-bit integer, the below flow is a guideline for sending a command to the card(s):

```

send_command(cmd_index, cmd_arg, other requirements)
{
WORD wCmd; // 32-bit integer to make up the data to write into transfer type register
(XFERTYP),
it is recommended to implement in a bit-field manner
wCmd = (<cmd_index> & 0x3f) >> 24; // set the first 8 bits as '00'+<cmd_index>
set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL accordind to the command index;
if (DMA is used) wCmd |= 0x1;
if (multi-block transfer) {
    set MSBSEL bit;
    if (finite block number) {
        set BCEN bit;
        if (auto12 command is to use) set AC12EN bit;
    }
}
write_reg(CMDARG, <cmd_arg>); // configure the command argument
write_reg(XFERTYP, wCmd); // set transfer type register (XFERTYP) as wCmd value to issue the
command
}
wait_for_response(cmd_index)
{
while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
read IRQ Status register and check if any error bits about Command are set
if (any error bits are set) report error;
write 1 to clear CC bit and all Command Error bits;
}

```

For the sake of simplicity, the function `wait_for_response` is implemented here using polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. By doing this, make sure the corresponding interrupt status bits are enabled.

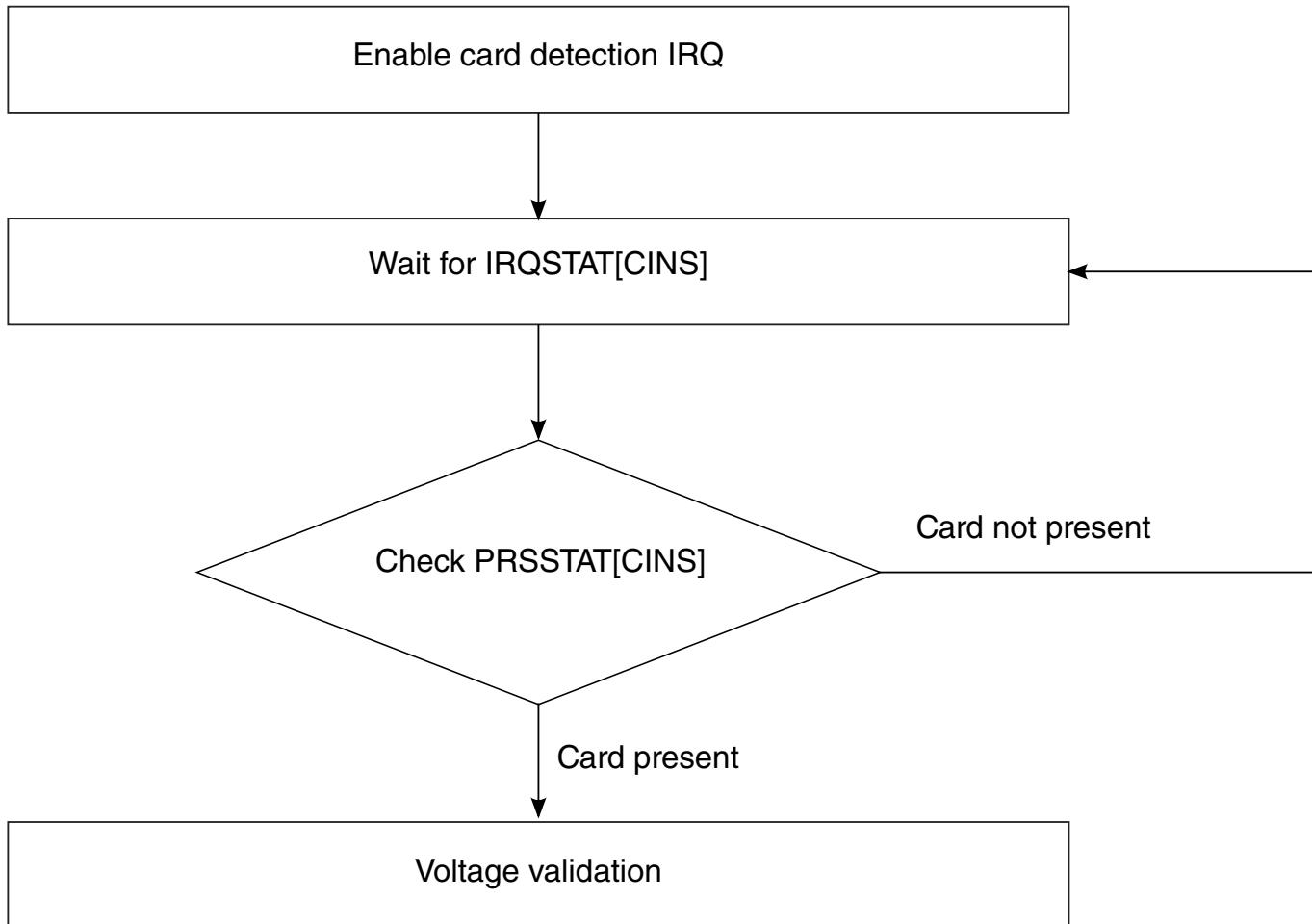
For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and go to the Standby State, no response to the Host when CMD2 is sent. The host driver should deal with 'fake' errors like this with caution.

## 22.5.2 Card identification mode

When a card is inserted to the socket or the card was reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the relative card address (RCA) or set the RCA for the MMC cards. All data communications in the card identification mode use the command line (CMD) only.

### 22.5.2.1 Card detect

The figure below illustrates a flow diagram showing the detection of MMC, SDIO, and SD cards using the eSDHC.



**Figure 22-19. Flow diagram for card detection**

The card detect sequence is as follows:

1. Set the IRQSTAT[CINSIEN] bit and IRQSIGEN[CINSIEN] bit to enable card detection interrupt.
2. When an interrupt from the eSDHC is received in IRQSTAT[CINS], check PRSSTAT[CINS] to confirm if it was caused by card insertion.

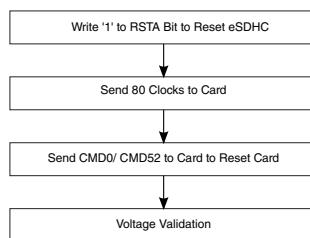
### 22.5.2.2 Reset

The host consists of three types of resets:

- Hardware reset (card and host) which is driven by POR (power on reset).

- Software reset (Host Only) is proceed by the write operation on SYSCTL[RSTD], SYSCTL[RSTC], or SYSCTL[RSTA] bits to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (card only). The command, "Go\_Idle\_State" (CMD0), is the software reset command for all types of MMC cards and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SD I/O Card, CMD52 is used to write an I/O reset in the CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. Figure below illustrates the software flow to reset both the eSDHC and the card.



**Figure 22-20. Flow chart for reset of the eSDHC and SD I/O card**

```

software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the Host
set DTOCV and SDCLKFS bit fields to get the SDHC_CLK of frequency around 400 KHz
configure IO pad to set the power voltage of external card to around 3.0V
poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
  
```

### 22.5.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification.

However, the supported minimum and maximum values for Vdd are defined in the operation conditions register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer Vdd conditions. This means, if the host and card have non-common Vdd ranges, the card will not be able to complete the identification cycle, nor will it be able to send CSD data.

Therefore, a special command Send\_Op\_Cont (CMD1 for MMC), SD\_Send\_Op\_Cont (ACMD41 for SD Memory) and IO\_Send\_Op\_Cont (CMD5 for SD I/O) is used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards which do not match the Vdd range(s) desired by the host. This is accomplished by the host sending the desired Vdd voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive State. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the inactive state. This query should be used if the host is able to select a common voltage range or if a notification should be sent to the system when a non-usuable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_arguement)
{
label the card as UNKNOWN;

send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
operation voltage, command argument is zero

if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
    if (0 < number of IO functions) {
        label the card as SDIO;
        IORDY = 0;
        while (!(IORDY in IO OCR response)) { // set voltage range for each IO
function
            send_command(IO_SEND_OP_COND, <voltage range>, <other
parameter>);
            wait_for_response(IO_SEND_OP_COND);
        } // end of while ...
    } // end of if (0 < ...
    if (memory part is present inside SDIO card) Label the card as SDCombo; // this is
an
SD-Combo card
} // end of if (RESP_TIMEOUT ...
if (the card is labelled as SDIO card) return; // card type is identified and voltage range
is
set, so exit the function;
send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific
CMD
prefix
if (no error calling wait_for_response(APP_CMD, <...>)) { // CMD55 is accepted
    send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage
range
for memory part or SD card
    wait_for_response(SD_APP_OP_COND); // voltage range is set
    if (card type is UNKNOWN) label the card as SD;
    return; //
} // end of if (no error ...
else if (errors other than time-out occur) { // command/response pair is corrupted
    deal with it by program specific manner;
} // of else if (response time-out
else { // CMD55 is refuse, it must be MMC card
    if (card is already labelled as SDCombo) { // change label
        re-label the card as SDIO;
        ignore the error or report it;
        return; // card is identified as SDIO card
    } // of if (card is ...
    send_command(SEND_OP_COND, <voltage range>, <...>);
    if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
```

```

either
    label the card as UNKNOWN;
    return;
} // of if (RESP_TIMEOUT ...
else label the card as MMC;
} // of else
}

```

#### 22.5.2.4 Card registry

Card registry for the MMC and SD/SDIO/SD combo cards are different.

For the SD card, the identification process starts at a clock rate lower than 400 KHz and the power voltage higher than 2.7 V (as defined by the card specification). At this time, the CMD line output drivers are push-pull drivers instead of open-drain. After the bus is activated, the host will request the card to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command should be sent to all of the new cards in the system. Incompatible cards are put into the inactive state. The host then issues the command, All\_Send\_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the identification state.

The host then issues Send\_Relative\_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA will be used to address the card for future data transfer operations. Once the RCA is received, the card changes its state to the standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send\_Relative\_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in system.

For MMC operation, the host starts the card identification process in open-drain mode with the identification clock rate lower than 400 KHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line allow parallel card operation during card identification. After the bus is activated the host will request the cards to send their valid operation conditions (CMD1). The response to CMD1 is the "wired OR" operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive State. The host then issues the broadcast command All\_Send\_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards (the cards in Ready State) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop

sending their CID immediately and must wait for the next identification cycle. Since the CID is unique for each card, only one card can be successfully send its full CID to the host. This card then goes into the Identification State. Thereafter, the host issues Set\_Relative\_Addr (CMD3) to assign to the card a relative card address (RCA). Once the RCA is received the card state changes to the Stand-by State, and the card does not react in further identification cycles, and its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

```

card_registry()
{
do { // decide RCA for each card until response time-out
    if(card is labelled as SDCombo or SDIO) { // for SDIO card like device
        send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to
publish its
RCA
        retrieve RCA from response;
    } // end if (card is labelled as SDCombo ...
    else if (card is labelled as SD) { // for SD card
        send_command(ALL_SEND_CID, <...>);
        if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
        send_command(SET_RELATIVE_ADDR, <...>);
        retrieve RCA from response;
    } // else if (card is labelled as SD ...
    else if (card is labelled as MMC) {
        send_command(ALL_SEND_CID, <...>);
        rca = 0x1; // arbitrarily set RCA, 1 here for example
        send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper
16
bits
    } // end of else if (card is labelled as MMC ...
} while (response is not time-out);
}

```

## 22.5.3 Card access

This section describes access to the card.

### 22.5.3.1 Clocking constraints

Table 22-11. Clocking Constraints

Mode name	Frequency	Source clock option	Minimum clock divisor	Maximum clock divisor
Default Speed	0-25 MHz	IPG clock, IPG PER clock	Card clock should not exceed 25 MHz.	No restriction
High Speed	0-50 MHz	IPG clock, IPG PER clock	Card clock should not exceed 50 MHz.	No restriction
SDR12	0-25 MHz	IPG clock, IPG PER clock	Card clock should not exceed 25 MHz.	No restriction
SDR25	0-50 MHz	IPG clock, IPG PER clock	Card clock should not exceed 50 MHz.	No restriction

Table continues on the next page...

**Table 22-11. Clocking Constraints (continued)**

Mode name	Frequency	Source clock option	Minimum clock divisor	Maximum clock divisor
eMMC DDR50	0-52 MHz	IPG clock, IPG PER clock	Card clock should not exceed 52 MHz. Odd clock division in DDR mode is supported only with IPG PER clock.	No restriction. Odd clock division in DDR mode is supported only with IPG PER clock.
DDR50	0-50 MHz	IPG clock, IPG PER clock	Card clock should not exceed 50 MHz. Odd clock division in DDR mode is supported only with IPG PER clock.	No restriction. Odd clock division in DDR mode is supported only with IPG PER clock.
SDR50 (without tuning)	0-100 MHz	IPG clock, IPG PER clock	Card clock should not exceed 100 MHz.	No restriction
SDR50 (with tuning)	0-100 MHz	IPG PER clock	3	16
SDR104	0-208 MHz	IPG PER clock	3	16
HS200	0-200 MHz	IPG PER clock	3	16
HS400	0-200 MHz	IPG PER clock	3	16

**NOTE**

IPG PER CLK is the clock selected through RCW[HWA\_CGA\_M2\_CLK\_SEL] and the IPG clock is platform clock.

### 22.5.3.2 Block write

#### 22.5.3.2.1 Normal write

During a block write (CMD24 - 27), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host.

If the CRC fails, the card should indicate the failure on the DAT line. The transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE\_BLK\_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS\_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write protected area.

For MMC and SD cards, programming of the CID and CSD registers do not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card will report an error and not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card will begin writing and hold the DAT line low if its write buffer is full and unable to accept new data from a new WRITE\_BLOCK command. The host may poll the status of the card with a SEND\_STATUS command (CMD13) or other means for SDIO cards at any time, and the card will respond with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby State and release the DAT line without interrupting the write operation. When re-selecting the card, it will reactivate the busy indication by pulling DAT to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card using DMA mode is as follows:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
  - a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
  - b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT (CMD52) to set the I/O block size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)
3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or some other error that occurred during the auto12 command sending and response receiving.

The software flow to write to a card using CPU Polling mode is as follows:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
  - a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
  - b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)

3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Issue the command with data transfer. The AC12EN bit should also be set.
6. Wait for IRQSTAT[BWR] interrupt to be set.
7. Write to Buffer port register for no. of times programmed in WML[WR\_WML] considering restriction mentioned in [Software polling procedure](#).
8. Clear IRQSTAT[BWR].
9. Repeat 6-8 steps for rest of the data transfer.
10. Wait for the Transfer Complete interrupt.
11. Check the status bit to see if a write CRC error occurred, or some other error that occurred during the auto12 command sending and response receiving.

### 22.5.3.2.2 Write with pause

The write operation can be paused during the transfer.

Instead of stopping the SDHC\_CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the PROCTL[SABGREQ] to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DAT0 line is not required to de-assert to release the busy state, no suspend command is needed.

Like in the flow described in [Normal write](#), the write with pause is shown with the same kind of write operation:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
  - a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
  - b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O block size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)
3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.

10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of BLKATTR[BLKCNT]. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The driver should read the value of BLKCNT after the transfer is paused and the transfer complete interrupt is received.

It is also possible the last block has begun when the stop at block gap request is sent to the buffer. In this case, the next block gap is actually the end of the transfer. These types of requests are ignored and the driver should treat this as a non-pause transfer and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this (if not needed), it is recommended to avoid using the suspend command for the SDIO card. This is because, when such a command is sent, the eSDHC thinks that the system will switch to another function on the SDIO card, and flush the data buffer. The eSDHC takes the resume command as a normal command with data transfer, and it is left for the driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, XFERTYP[MSBSEL] and XFERTYP[BCEN] are set as well as XFERTYP[AC12EN]. However, the eSDHC will automatically send a CMD12 to mark the end of the multi-block transfer.

### **22.5.3.3 Block read**

#### **22.5.3.3.1 Normal read**

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification.

A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the transfer state. For multi-blocks read, data blocks will be continuously transferred until a stop command is issued.

The software flow to read from a card using DMA mode is as follows:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:

- a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
- b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O block size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)
3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer read ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or some other error, occurred during the auto12 command sending and response receiving.

The software flow to read from a card using CPU polling mode is as follows:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
  - a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
  - b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O block size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)
3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Issue the command with data transfer. The AC12EN bit should also be set.
6. Wait for IRQSTAT[BRR] interrupt to be set.
7. Read from Buffer port register for number of times programmed in WML[RD\_WML] considering restriction mentioned in [Software polling procedure](#).
8. Clear IRQSTAT[BRR].
9. Repeat 6-8 steps for rest of the data transfer.
10. Wait for the Transfer Complete interrupt.
11. Check the status bit to see if a read CRC error occurred, or some other error, occurred during the auto12 command sending and response receiving.

### **22.5.3.3.2 Read with pause**

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the read wait feature can pause during the read operation.

If the SDIO card support read wait (SRW bit in CCCR register is 1), the driver can set the PROCTL[SABGREQ] to pause the transfer between the data blocks. Before setting the SABGREQ bit, make sure the PROCTL[RWCTL] is set, otherwise the eSDHC will not assert the read wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit once the read wait capability of the SDIO card is recognized.

Like in the flow described in [Normal read](#), the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm the card supports read wait.
2. Set the RWCTL bit.
3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
  - a. For SD/MMC cards , use SET\_BLOCKLEN (CMD16)
  - b. For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O block size bit field in the CCCR register (for function 0) or FBR register (for functions 1-7)
5. Set the eSDHC block length register to be the same as the block length set for the card in Step 4.
6. Set the eSDHC number block register (NOB), nob is 5 (for instance).
7. Disable the buffer read ready interrupt, configure the DMA setting and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the transfer complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the transfer complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or some other error, occurred during the auto12 command sending and response receiving.

Like the write operation, it is possible to meet the ending block of the transfer when paused. In this case, the eSDHC will ignore the stop at block gap request and treat it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause will be transferred to the Host System. Even if the Suspend Command is sent or not, the internal data buffer is not flushed.

If the Suspend Command is sent and the transfer is later resumed by means of a Resume Command, the eSDHC takes the command as a normal one accompanied with data transfer. It is left for the driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, XFERTYP[MSBSEL] and XFERTYP[BCEN] are set, as well as XFERTYP[AC12EN]. However, the eSDHC will automatically send the CMD12 to mark the end of multi-block transfer.

### 22.5.3.4 Suspend resume

The eSDHC supports the suspend resume operations of SDIO cards, although slightly different than the suggested implementation of suspend in the SDIO card specification.

#### 22.5.3.4.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of the SDIO card. The eSDHC does not monitor the content of the response, so it doesn't know if the Suspend command succeeded or not.

Accordingly, it doesn't de-assert Read Wait for read pause. To solve this problem, the driver should not mark the Suspend command as a "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver should send this command as if it were a normal command, and only when the command succeeds, and the BS bit is set in the response, can the driver send another command marked as "Suspend" to inform the eSDHC that the current transfer is suspended. As shown in the following sequence for Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, save the context registers in the system memory for later use, including the SDMA system address register (DSADDR) for DMA operation, and the block attributes register (BLKATTR).
3. Send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
4. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the driver strategy.
5. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so the eSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, the eSDHC stops driving DAT2 and goes to the idle state.
6. Begin operation for another function on the SDIO card.

### 22.5.3.4.2 Resume

To resume the data transfer, a resume command should be issued:

1. To resume the suspended function, restore the context register with the saved value in step #2 of the suspend operation above.
2. Send the resume command. In the transfer type register (XFERTYP), all bit fields are set to the value as if this were another ordinary data transfer, instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the resume command has responded, the data transfer will be resumed.

### 22.5.3.5 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command.

To accomplish this:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4 Kbyte address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1-3.
4. Repeat steps 1-3 until all descriptors are created.
5. In the last descriptor, set the end flag to 1 and make sure the total length of all descriptors match the product of the block size and block number configured in the block attributes register (BLKATTR).
6. Set the ADMA system address register (ADSADDR) to the address of the first descriptor and set PROCTL[DMAS] to 01 to select the ADMA.
7. Issue a write or read command with XFERTYP[DMAEN] set.

Steps 1-5 are independent of step 6, so step 6 can finish before steps 1-5. Regarding the descriptor configuration, it is recommended not to use the link descriptor as it requires extra system memory access.

### 22.5.3.6 Tuning block procedure

Tuning block is used for SD UHS SDR50, SDR104 and MMC HS200 modes.

Host driver should execute tuning procedure before initiating data transfer with these speed modes.

The steps for using tuning block are:

1. Clear SYSCTL[SDCLKEN]
2. Wait for PRSSTAT[SDSTB] to be set
3. Set ESDHCCTL[FAF]
4. Wait for ESDHCCTL[FAF] to be cleared
5. Set appropriate AUTOCERR[UHSM]
6. Set TBCTL[TB\_EN] and program appropriate TBCTL[TB\_MODE]
7. Set SYSCTL[SDCLKEN]
8. Wait for PRSSTAT[SDSTB] to be set
9. Execute tuning procedure

While tuning procedure is being performed, eSDHC does not generate any other command or data interrupt except buffer read ready in IRQSTAT register.

When tuning error is received, host driver should abort the current data transfer and execute the tuning procedure.

ESDHCCCTL[PCS] (peripheral clock select), and TBCTL[TB\_EN] (tuning block enable) should always be set whenever using tuning block. PCS should be set before TB\_EN.

#### **NOTE**

- Similar steps needs to be performed to disable tuning block, except programming different values in [5](#) and [6](#). Also [9](#) need not to be performed.
- For tuning mode operation, the SD clock divisor value must be within 3 to 16.
- Tuning data timeout occur with default value of SYSCTL[DTOCV] when SD card send tuning data after 450  $\mu$ s. According to SD host controller specification a card may take up to 150 ms (or at least 100 ms) to respond to the first tuning block.

#### **22.5.3.6.1 Tuning procedure for hardware tuning modes**

The steps for tuning procedure, when TBCTL[TB\_MODE] is programmed to either Mode 1, Mode 2 or Mode 3, are:

1. Set SYSCTL2[EXTN], execute tuning.
2. Issue SEND\_TUNING\_BLK Command (CMD19 for SD, CMD21 for MMC).
3. Wait for IRQSTAT[BRR], buffer read ready, to be set.
4. Clear IRQSTAT[BRR].
5. Check SYSCTL2[EXTN] to be cleared.
6. Repeat steps 2-5, if EXTN is not cleared.

7. Check SYSCTL2[SMPCLKSEL], sampling clock select. It's set value indicates tuning procedure success, and clears indicate failure. In case of tuning failure, fixed sampling scheme could be used by clearing TBCTL[TB\_EN], tuning block enable bit.

### 22.5.3.6.2 Tuning procedure for software tuning mode

The steps for tuning procedure, when TBCTL[TB\_MODE] is set to SW tuning mode, are:

1. Program the start and end pointer for the data window in the TPR register.
2. Set SYSCTL2[EXTN] and SYSCTL2[SMPCLKSEL], execute tuning.
3. Issue SEND\_TUNING\_BLK Command (CMD19 for SD, CMD21 for MMC).
4. Wait for IRQSTAT[BRR], buffer read ready, to be set.
5. Clear IRQSTAT[BRR].
6. Check SYSCTL2[EXTN] to be cleared.
7. Check SYSCTL2[SMPCLKSEL], sampling clock select. It's set value indicates tuning procedure success, and clears indicate failure. In case of tuning failure, fixed sampling scheme could be used by clearing TBCTL[TB\_EN], tuning block enable bit.

### 22.5.3.7 DDR

The steps for using DDR mode are:

1. Clear SYSCTL[SDCLKEN]
2. Wait for PRSSTAT[SDSTB] to be set
3. Program AUTOCERR[UHSM] to 4
4. If required, change the clock division ratio in SYSCTL register
5. Set SDCLKCTL[CMD\_CLK\_CTL] and SDCLKCTL[LPBK\_CLK\_SEL] for DDR mode
6. Set SYSCTL[SDCLKEN]
7. Wait for PRSSTAT[SDSTB] to be set
8. Again clear SYSCTL[SDCLKEN]
9. Wait for PRSSTAT[SDSTB] to be set
10. Set ESDHCCTL[FAF]
11. Wait for ESDHCCTL[FAF] to be cleared
12. Set SYSCTL[SDCLKEN]
13. Wait for PRSSTAT[SDSTB] to be set

**NOTE**

1. SD clock divisor should be even for DDR mode.
2. Similar steps needs to be performed to disable, except programming different value in [3](#)

**22.5.3.8 Transfer error****22.5.3.8.1 CRC transfer error**

It is possible at the end of a block transfer, that a write CRC status error or read CRC error occurs.

For this type of error the latest block received should be discarded.

This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one. For a multi-block transfer, the host driver should issue a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, the eSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 will be sent by the eSDHC. In this case, the driver should re-send or re-obtain the last block with a single block transfer.

**22.5.3.8.2 DMA transfer error**

During the data transfer with internal single DMA, if the DMA engine encounters some error on the System bus, the DMA operation is aborted and DMA error interrupt is sent to the host system.

When acknowledged by such an interrupt, the driver should calculate the start address of data block in which the error occurs. The start address can be calculated by:

Read the DMA error address register (DMAERRADDR). Taking the block size and the start address intially preprogrammed in SDMA Adress register, it is straight forward to obtain the start address of the corrupted block.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and re-start the transfer from the corrupted block to recover from the error.

### 22.5.3.8.3 ADMA transfer error

There are three kinds of possible ADMA errors; The System transfer, invalid descriptor, and data-length mismatch errors.

Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set. For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

1. System transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or the next block if no block is corrupted.
2. Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and re-create the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
3. Data-length mismatch error: It is similar to recover from this error. The host driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain, and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

### 22.5.3.8.4 Auto CMD12 error

After the last block of the multi block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, the eSDHC automatically sends a CMD12 to the card to stop the transfer.

When errors with this command occur, it is recommended to the driver to deal with the situations in the following manner:

1. Auto CMD12 response time-out. It is not certain whether the command is accepted by the card or not. The driver should clear the Auto CMD12 error status bits and re-send the CMD12 until it is accepted by the card.
2. Auto CMD12 response CRC error. Since card responds to the CMD12, the card will abort the transfer. The driver may ignore the error and clear the error status bit.
3. Auto CMD12 conflict error or not sent. The command is not sent, so the driver should send a CMD12 manually.

### 22.5.3.9 Card interrupt

The external cards can inform the host controller by means of some special signals. For the SDIO card, it can be the low level on the DAT[1] line during some special period.

The eSDHC only monitors the DAT[1] line and supports the SDIO interrupt.

When the SDIO interrupt is captured by the eSDHC, and the Host System is informed by the eSDHC asserting the eSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt factor is controlled by the external card, the interrupt from the SDIO card must be served before the CINT bit is cleared by written 1. Refer to [Card interrupt handling](#) for the card interrupt handling flow.

### 22.5.4 Switch function

MMC cards transferring data at bus widths other than 1-bit is a new feature added to the MMC spec.

The high speed timing mode for all card devices, was also recently defined in various card specifications. To enable these new features, a "switch" command should be issued by the host driver.

For SDIO cards, the high speed mode is enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high speed mode is queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH\_FUNC). For MMC cards, the high speed mode is queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The 4-bit and 8-bit bus width of the MMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO cards, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following flowcharts do not show current capability check, which is recommended in the function switch process.

### 22.5.4.1 Query, enable and disable SDIO high speed mode

```

enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report the SDIO card does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is
cleared;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}

```

### 22.5.4.2 Query, enable and disable SD high speed mode

```

enable_sd_high_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFF1 and read 64 bytes of data accompanying the R1 response;
wait data transfer done bit is set;
check if the bit 401 of received 512 bit is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
send CMD6, with argument 0x80FFFFF1 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sd_high_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}

```

### 22.5.4.3 Query, enable and disable MMC high speed mode

```

enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high speed mode and
return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
}

```

```

check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}

```

## 22.5.4.4 Set MMC bus width

```

change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and
return;
send CMD6 with argument 0x3B70x00; (8-bit, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}

```

## 22.5.5 ADMA operation

### 22.5.5.1 ADMA1 operation

```

Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB align);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}

```

```
}

if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

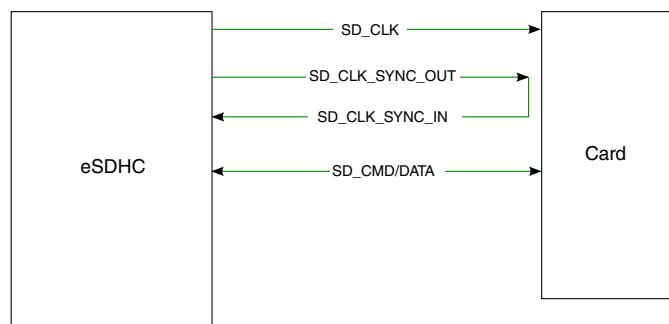
### 22.5.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}
```

## 22.6 Interfacing Card

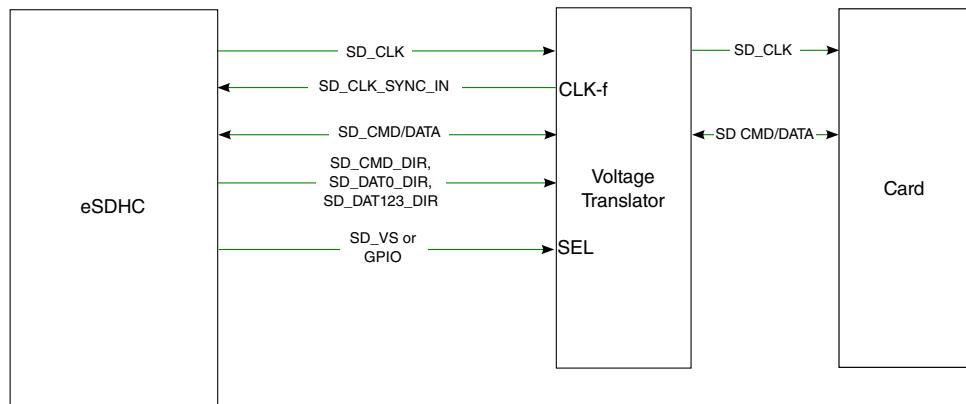
eSDHC and card IO voltage might be different. External on board voltage translator can be used to interface the card in that case.

Following diagram illustrates direct interfacing of eSDHC with card when voltage level is same at both ends. SD\_CLK\_SYNC\_OUT and SD\_CLK\_SYNC\_IN should be routed as close as possible to card, with minimum skew with respect to SD\_CLK.



**Figure 22-21. Direct Interfacing of eSDHC with Card**

Below figure illustrates interfacing card with eSDHC through voltage translator when the voltages are different. CMD/DATA DIR and SD\_VS pins might be required when interfacing with voltage translator that support DDR, and SDR more than 50 MHz modes. eSDHC SD\_VS, or GPIO could be used to control SEL pin of the translator.



**Figure 22-22. Card Interfacing with eSDHC through Voltage Translator**

#### NOTE

CD, WP and DAT4-7 are not shown in above figures for simplicity.

## 22.7 Commands for MMC/SD/SDIO and

See the table below for the list of commands for the MMC/SD/SDIO cards .

Refer the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the MultiMediaCard:

- Broadcast commands (bc), no response
- Broadcast commands with response (bcr), response from all cards simultaneously
- Addressed (point-to-point) commands (ac), no data transfer on the DAT
- Addressed (point-to-point) data transfer commands (adtc)

The access bits for the EXT\_CSD access modes are shown in [Table 22-13](#).

**Table 22-12. Commands for MMC/SD/SDIO cards**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.

*Table continues on the next page...*

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3 <sup>1</sup>	ac	[31:16] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADD_R	Assigns relative address to the card.
CMD4	bc	[31:16] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD6 <sup>2</sup>	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 - 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer "SD Physical Specification V1.1" for more details.
CMD6 <sup>3</sup>	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer "The MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8 <sup>4</sup>	bcr	[31:12] reserved bits [11:8] supply voltage (VHS) [7:0] check pattern	R7	SEND_IF_COND	Sends SD memory card interface condition, which includes host supply voltage information and asks the card whether card supports voltage. Reserved bits shall be set to 0.
CMD8 <sup>5</sup>	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.

*Table continues on the next page...*

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
CMD9	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	ac	[31:0] stuff bits	R1	VOLTAGE_SWITCH	Switch to 1.8V bus signalling level. Applicable for UHS SD card.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:16] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14		Reserved			
CMD15	ac	[31:16] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	adtc	[31:0] stuff bits	R1	SEND_TUNING_BLOCK	64 bytes tuning pattern is sent for SDR50 and SDR104. Applicable for UHS SD card.
CMD20		Reserved			
CMD21	adtc	[31:0] stuff bits	R1	SEND_TUNING_BLOCK	128 clocks of tuning pattern (64 byte in 4-bit mode or 128 byte in 8-bit mode) is sent for HS200 optimal sampling point detection. Applicable for MMC card.
CMD22		Reserved			
CMD23	ac	[31:16] set to 0 [15:0] block count	R1	SET_BLOCKCOUNT	Defines the number of blocks which are going to be transferred in the immediately succeeding multiple block read or write command. If the argument is all 0s, the subsequent read/write operations are openedended.
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command should be issued only once per card. The card contains hardware to prevent this

*Table continues on the next page...*

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
					operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31				Reserved	
CMD32	ac	[31:0] data address	R1	ERASE_WR_BLK_START	Sets the address of the first write block to be erased in SD card.
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group in MMC card.
CMD33	ac	[31:0] data address	R1	ERASE_WR_BLK_END	Sets the address of the last write block of the continuous range to be erased in SD card.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase in MMC card.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection in MMC card.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase in MMC card.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase in MMC card.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection in MMC card.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This

Table continues on the next page...

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
					command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode in MMC card.
CMD41				Reserved	
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43-51				Reserved	
CMD52	ac	As per SDIO spec CMD52 format	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	As per SDIO spec CMD53 format	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54				Reserved	
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-63				Reserved	
ACMD6	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD18 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SECURE_READ_MULT_I_BLOCK	Protected Area Access Command: Reads continuously transfer data blocks from protected area of SD memory card. Refer Security Specification Version 2.00 for more details.
ACMD22 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SEC_TORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.
ACMD23 <sup>6</sup>	ac	[31:0] stuff bits	R1	SET_WR_BLK_ERASE_COUNT	In SD cards, set the number of write blocks to be pre-erased before writing (to be used for faster multiple block write command)

*Table continues on the next page...*

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
					Default value is 1 (one write block).
ACMD25 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SECURE_WRITE_MUL_TI_BLOCK	Protected Area Access Command: Writes continuously transfer data blocks to protected area of SD memory card. Refer Security Specification Version 2.00 for more details.
ACMD26 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SECURE_WRITE_MKB	System Area Access Command: Overwrite the existing media key block (MKB) on the system area of SD Memory Card with new MKB. This command is used in dynamic update MKB scheme. Refer Security Specification Version 2.00 for more details.
ACMD38 <sup>6</sup>	ac	[31:0] stuff bits	R1b	SECURE_ERASE	Protected Area Access Command: Erase a specified region of the Protected Area of SD Memory Card. Refer Security Specification Version 2.00 for more details.
ACMD41 <sup>6</sup>	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42 <sup>6</sup>	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DET_ECT	Connect [1]/Disconnect [0] the 50KΩ pull-up resistor on CD/DAT3 of SD card.
ACMD43 <sup>6</sup>	adtc	[31:24]Unit_Count: [23:16] MKB_ID: [15:0]Unit_Offset:	R1	GET_MKB	Reads Media Key Block from the System Area of SD Memory Card. -Unit_Count specifies the Number of units to read. (Here, a unit=512 byte (fixed).) - MKB_ID specifies the application unique number. - Unit_Offset specifies the start address(offset) to read. Refer Security Specification Version 2.00 for more details.
ACMD44 <sup>6</sup>	adtc	[31:0] stuff bits	R1	GET_MID	Reads Media ID from the system area of SD memory card. Refer Security Specification Version 2.00 for more details.
ACMD45 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SET_CER_RN1	AKE Command: Writes random number RN1 as challenge1 in AKE process. Refer Security Specification Version 2.00 for more details.
ACMD46 <sup>6</sup>	adtc	[31:0] stuff bits	R1	GET_CER_RN2	AKE Command:

*Table continues on the next page...*

**Table 22-12. Commands for MMC/SD/SDIO cards (continued)**

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
					Reads random number RN2 as challenge2 in AKE process. Refer security specification version 2.00 for more details.
ACMD47 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SET_CER_RES2	AKE Command: Writes RES2 as response2 to RN2 in AKE process Refer Security Specification Version 2.00 for more details.
ACMD48 <sup>6</sup>	adtc	[31:0] stuff bits	R1	GET_CER_RES1	AKE Command: Reads RES1 as response1 to RN1 in AKE process. Refer Security Specification Version 2.00 for more details.
ACMD49 <sup>6</sup>	ac	[31:0] stuff bits	R1b	CHANGE_SECURE_AREA	Protected Area Access Command: Change size of the protected area. Refer Security Specification Version 2.00 for more details.
ACMD51 <sup>6</sup>	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD configuration register (SCR).

1. CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET\_RELATIVE\_ADDR, with a response type of R1. For SD cards, it is referred to as SEND\_RELATIVE\_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed MMC cards and high speed SD cards. Command SWITCH\_FUNC is for high speed SD cards.
3. Command SWITCH is for high speed MMC cards. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH\_ERROR status bit in the EXT\_CSD register is set. The Access Bits are shown in [Table 2](#).
4. CMD8 for SD stands for SEND\_IF\_COND.
5. CMD8 for MMC stands for SEND\_EXT\_CSD.
6. ACMDs should be preceded with the APP\_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).

**Table 22-13. EXT\_CSD access modes**

Bits	Access name	Operation
00	Command set	The command set is changed according to the Cmd set field of the argument.
01	Set Bits	The bits in the pointed byte are set, according to the 1 bit in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the 1 bit in the Value field.
11	Write Byte	The Value field is written into the pointed byte.

## 22.8 Software restrictions

This section covers software restrictions.

### 22.8.1 Software polling procedure

For polling read or write, once the software begins a buffer read or write, it must access the buffer data port register (DATPORT) exactly the number of times as watermark level value set in the watermark level register (WML).

However, if the block size is not same as of the value in watermark level register (read and write respectively), the software must access exactly the remaining number of words at the end of each block. For example, for read operations, if the RD\_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block count is 2, then the access times to data buffer port register for the burst sequence in the whole transfer process must be 4, 4, 2 (for first block); 4, 4, 2 (for second block).

### 22.8.2 Suspend operation

In order to suspend the data transfer, the software must inform eSDHC that the suspend command is successfully accepted.

To achieve this, after the suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as "01") to inform eSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remained number of blocks before sending the normal command marked as suspend, otherwise on sending such "suspend" command, eSDHC will regard the current transfer a aborted and change the BLKCNT register to its original value, instead of keeping the remaining number of blocks.

### 22.8.3 Data port access

Data port does not support parallel access.

For example, during an external DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or external DMA. Otherwise, the data would be corrupted inside the eSDHC buffer.

## 22.8.4 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for MMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or SD security read commands, or whatever multi-block read without abort command at card side; soft reset for data(SYSCTL[RSTD]) is required after the transfer is complete to drive the internal state machine to idle mode.

## 22.8.5 ADMA address

For ADMA1/ADMA2 operation, the ADMA system address register (ADSADDR) and address defined in the address field of the descriptor table should be 4-byte aligned.

## 22.8.6 Allowed operations after stop at block gap

Only one of these operations is allowed after stop at block gap event:

- Continue data transfer by setting SYSCTL[CREQ]
- Issue suspend command
- Abort data transfer by issued abort command

No other command can be issued until one of the operations listed above is performed.

## 22.8.7 SDIO card interrupt during soft reset

The host driver should disable SDIO card interrupts in IRQSTAT[CINT] before issuing soft reset for data or all (generally used for error recovery) in the system control register (SYSCTL), and enable it after error recovery sequence has been completed.

## 22.8.8 Soft reset for data not allowed when SD clock is disabled

Soft reset for data and CMD (SYSCTL[RSTD]/SYSCTL[RSTC]) should not be issued when SD clock is disabled; that is, when SYSCTL[SDCLKEN] is cleared.

Instead, the host driver may issue soft reset for all (SYSCTL[RSTA]).

## 22.8.9 Data transfer with Auto CMD12 Enable

When Auto CMD12 is enabled for data transfer, it generates IRQSTAT[TC] for data transfer completion but does not generate TC for Auto CMD12(command with busy).

Host Driver needs make sure that card has reached to “trans” state before issuing any new data command. CMD13(SEND\_STATUS) could be sent to check the card status.

# Chapter 23

## FlexTimer Module (FTM)

### 23.1 The FlexTimer module as implemented on the chip

This section provides details about how the FlexTimer module is implemented on the chip.

#### 23.1.1 LS1043A FlexTimer module integration

The following table describes the FlexTimer module integration into this chip:

**Table 23-1. FlexTimer module integration**

Module	Module Base address
FlexTimer Module 1	29D_0000
FlexTimer Module 2	29E_0000
FlexTimer Module 3	29F_0000
FlexTimer Module 4	2A0_0000
FlexTimer Module 5	2A1_0000
FlexTimer Module 6	2A2_0000
FlexTimer Module 7	2A3_0000
FlexTimer Module 8	2A4_0000

The remainder of this chapter refers to a single FlexTimer module. Notes are included to indicate variations for multiple instantiations.

#### 23.1.2 LS1043A FlexTimer signals

The following table lists the SoC signal names and their corresponding FlexTimer module signal names used in this chapter:

**Table 23-2. FlexTimer signals**

LS1043A signal name	FlexTimer module signal
FTMn_EXTCLK	EXTCLK
FTMn_CHn	CHn
FTMn_FAULT	FAULTj
FTMn_QD_PHA	PHA
FTMn_QD_PHB	PHB

### 23.1.3 LS1043A FlexTimer module special consideration

The FlexTimer module implements the following parameter settings in the chip:

**Table 23-3. LS1043A FlexTimer parameter settings**

FlexTimer parameters	LS1043A parameter value							
	FTM1	FTM2	FTM3	FTM4	FTM5	FTM6	FTM7	FTM8
Number of channels available at device I/O level	8	8	8	8	2	2	2	2
Quadrature decoding support	Yes	Yes	Yes	Yes	No	No	No	No
EXT_CLK support	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Fixed frequency clock support	32 KHz RTC	32 KHz RTC	32 KHz RTC	32 KHz RTC	32 KHz RTC	32 KHz RTC	32 KHz RTC	32 KHz RTC
System clock support	Platform clock/1	Platform clock/1	Platform clock/1	Platform clock/1	Platform clock/1	Platform clock/1	Platform clock/1	Platform clock/1
Stop mode support	Yes. Refers to the LPM20 low power mode of the chip.							

The table below shows the FlexTimer chaining for 32-bit counter. See [FTM chain configuration \(SCFG\\_FTM\\_CHAIN\\_CONFIG\)](#) for more details.

**Table 23-4. FlexTimer chaining for 32-bit counter**

FlexTimer-A	FlexTimer-B	Control bit	Control bit default value
FTM1	FTM5	SCFG_FTM_RESET[FTMCHN1]	0 = Chaining disabled
FTM2	FTM6	SCFG_FTM_RESET[FTMCHN2]	0 = Chaining disabled
FTM3	FTM7	SCFG_FTM_RESET[FTMCHN3]	0 = Chaining disabled
FTM4	FTM8	SCFG_FTM_RESET[FTMCHN4]	0 = Chaining disabled

It is possible to chain two FlexTimer modules to get a bigger 32-bit counter. This is achieved by:

- Connecting CH7 output of FlexTimer-B to PHA input of FlexTimer-A.

- Tying PHB input of FlexTimer-A to one.
- Programming FlexTimer-A to be in quad-mode.

FlexTimer-A has [31:16] of the 32-bit counter while FlexTimer-B has [15:0] of the 32-bit counter. The above table also shows how the chaining is implemented for FlexTimer-A and FlexTimer-B.

## 23.2 Introduction

The FlexTimer module (FTM) is a two-to-eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

### NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

### 23.2.1 FlexTimer philosophy

The FlexTimer is built upon a simple timer, the Timer PWM Module – TPM, used for many years on our HCS08 family of 8-bit microcontrollers. The FlexTimer extends the functionality to meet the demands of motor control, digital lighting solutions, and power conversion, while providing low cost and backwards compatibility with the TPM module.

Several key enhancements are made:

- Signed up counter
- Deadtime insertion hardware
- Fault control inputs
- Enhanced triggering functionality
- Initialization and polarity control

All of the features common with the TPM have fully backwards compatible register assignments. The FlexTimer can also use code on the same core platform without change to perform the same functions.

Motor control and power conversion features have been added through a dedicated set of registers and defaults turn off all new features. The new features, such as hardware deadtime insertion, polarity, fault control, and output forcing and masking, greatly reduce loading on the execution software and are usually each controlled by a group of registers.

FlexTimer input triggers can be from comparators, ADC, or other submodules to initiate timer functions automatically. These triggers can be linked in a variety of ways during integration of the sub modules so please note the options available for used FlexTimer configuration.

More than one FlexTimers may be synchronized to provide a larger timer with their counters incrementing in unison, assuming the initialization, the input clocks, the initial and final counting values are the same in each FlexTimer.

All main user access registers are buffered to ease the load on the executing software. A number of trigger options exist to determine which registers are updated with this user defined data.

## **23.2.2 Features**

The FTM features include:

- FTM source clock is selectable.
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the system clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In Input Capture mode:
  - The capture can occur on rising edges, falling edges or both edges
  - An input filter can be selected for some channels
- In Output Compare mode the output signal can be set, cleared, or toggled on match

- All channels can be configured for center-aligned PWM mode
- Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal
- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs
- The deadtime insertion is available for each complementary pair
- Generation of match triggers
- Initialization trigger
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- Synchronized loading of write buffered FTM registers
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input captures for a stuck at zero and one conditions
- Dual edge capture for pulse and period width measurement
- Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on external event

### 23.2.3 Modes of operation

When the chip is in an active mode, the FTM temporarily suspends all counting until the chip returns to normal user operating mode. During Stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During Wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the chip from Wait mode, the power can then be saved by disabling FTM functions before entering Wait mode.

## 23.2.4 Block diagram

The FTM uses one input/output (I/O) pin per channel, CH<sub>n</sub> (FTM channel (n)) where n is the channel number (0–7).

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

### NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

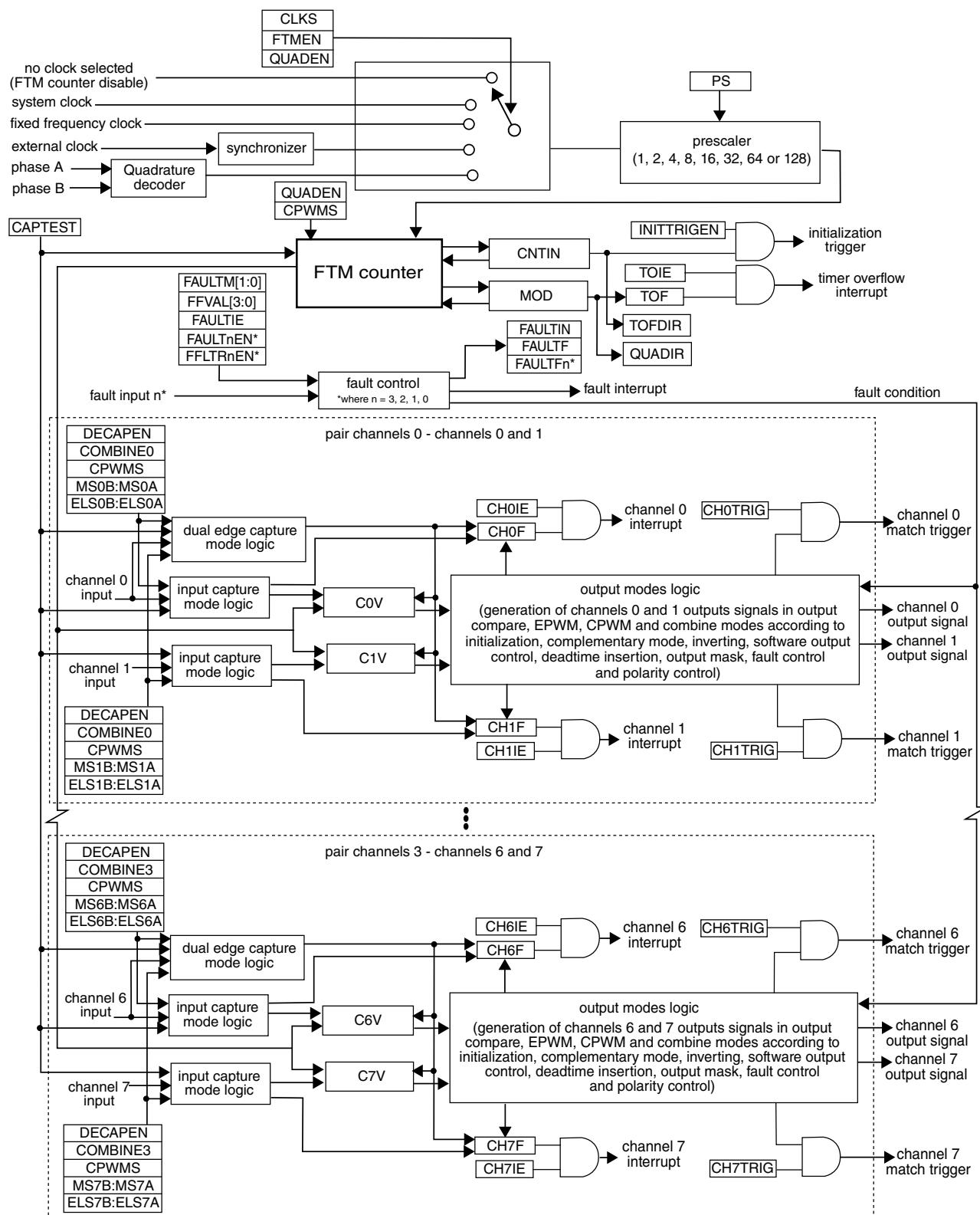


Figure 23-1. FTM block diagram

## 23.3 FTM signal descriptions

[Table 23-5](#) shows the user-accessible signals for the FTM.

**Table 23-5. FTM signal descriptions**

Signal	Description	I/O	Function
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I	The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of system clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.
CHn	FTM channel (n), where n can be 7-0	I/O	Each FTM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.
FAULTj	Fault input (j), where j can be 3-0	I	The fault input signals are used to control the CHn channel output state. If a fault is detected, the FAULTj signal is asserted and the channel output is put in a safe state. The behavior of the fault logic is defined by the FAULTM[1:0] control bits in the MODE register and FAULTEN bit in the COMBINEm register. Note that each FAULTj input may affect all channels selectively since FAULTM[1:0] and FAULTEN control bits are defined for each pair of channels. Because there are several FAULTj inputs, maximum of 4 for the FTM module, each one of these inputs is activated by the FAULTjEN bit in the FLTCTRL register.
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I	The quadrature decoder phase A input is used as the Quadrature Decoder mode is selected. The phase A input signal is one of the signals that control the FTM counter increment or decrement in the <a href="#">Quadrature Decoder mode</a> .
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I	The quadrature decoder phase B input is used as the Quadrature Decoder mode is selected. The phase B input signal is one of the signals that control the FTM counter increment or decrement in the <a href="#">Quadrature Decoder mode</a> .

## 23.4 Memory map and register definition

### 23.4.1 Memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

The registers and bits of an unavailable function in the FTM remain in the memory map and in the reset value, but they have no active function.

**Note**

Do not write in the region from the CNTIN register through the PWMLOAD register when FTMEN = 0.

**NOTE**

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

## 23.4.2 Register descriptions

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved.

**FTM memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
29D_0000	Status And Control (FTM1_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
29D_0004	Counter (FTM1_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
29D_0008	Modulo (FTM1_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
29D_000C	Channel (n) Status And Control (FTM1_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0010	Channel (n) Value (FTM1_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_0014	Channel (n) Status And Control (FTM1_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0018	Channel (n) Value (FTM1_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_001C	Channel (n) Status And Control (FTM1_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0020	Channel (n) Value (FTM1_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_0024	Channel (n) Status And Control (FTM1_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0028	Channel (n) Value (FTM1_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_002C	Channel (n) Status And Control (FTM1_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0030	Channel (n) Value (FTM1_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_0034	Channel (n) Status And Control (FTM1_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0038	Channel (n) Value (FTM1_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_003C	Channel (n) Status And Control (FTM1_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0040	Channel (n) Value (FTM1_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_0044	Channel (n) Status And Control (FTM1_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29D_0048	Channel (n) Value (FTM1_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29D_004C	Counter Initial Value (FTM1_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>

*Table continues on the next page...*

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
29D_0050	Capture And Compare Status (FTM1_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
29D_0054	Features Mode Selection (FTM1_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
29D_0058	Synchronization (FTM1_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
29D_005C	Initial State For Channels Output (FTM1_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
29D_0060	Output Mask (FTM1_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
29D_0064	Function For Linked Channels (FTM1_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
29D_0068	Deadtime Insertion Control (FTM1_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
29D_006C	FTM External Trigger (FTM1_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
29D_0070	Channels Polarity (FTM1_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
29D_0074	Fault Mode Status (FTM1_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
29D_0078	Input Capture Filter Control (FTM1_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
29D_007C	Fault Control (FTM1_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
29D_0080	Quadrature Decoder Control And Status (FTM1_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
29D_0084	Configuration (FTM1_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
29D_0088	FTM Fault Input Polarity (FTM1_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
29D_008C	Synchronization Configuration (FTM1_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
29D_0090	FTM Inverting Control (FTM1_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
29D_0094	FTM Software Output Control (FTM1_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
29D_0098	FTM PWM Load (FTM1_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
29E_0000	Status And Control (FTM2_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
29E_0004	Counter (FTM2_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
29E_0008	Modulo (FTM2_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
29E_000C	Channel (n) Status And Control (FTM2_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0010	Channel (n) Value (FTM2_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_0014	Channel (n) Status And Control (FTM2_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>

Table continues on the next page...

**FTM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
29E_0018	Channel (n) Value (FTM2_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_001C	Channel (n) Status And Control (FTM2_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0020	Channel (n) Value (FTM2_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_0024	Channel (n) Status And Control (FTM2_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0028	Channel (n) Value (FTM2_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_002C	Channel (n) Status And Control (FTM2_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0030	Channel (n) Value (FTM2_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_0034	Channel (n) Status And Control (FTM2_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0038	Channel (n) Value (FTM2_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_003C	Channel (n) Status And Control (FTM2_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0040	Channel (n) Value (FTM2_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_0044	Channel (n) Status And Control (FTM2_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29E_0048	Channel (n) Value (FTM2_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29E_004C	Counter Initial Value (FTM2_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
29E_0050	Capture And Compare Status (FTM2_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
29E_0054	Features Mode Selection (FTM2_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
29E_0058	Synchronization (FTM2_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
29E_005C	Initial State For Channels Output (FTM2_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
29E_0060	Output Mask (FTM2_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
29E_0064	Function For Linked Channels (FTM2_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
29E_0068	Deadtime Insertion Control (FTM2_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
29E_006C	FTM External Trigger (FTM2_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
29E_0070	Channels Polarity (FTM2_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
29E_0074	Fault Mode Status (FTM2_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
29E_0078	Input Capture Filter Control (FTM2_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
29E_007C	Fault Control (FTM2_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
29E_0080	Quadrature Decoder Control And Status (FTM2_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
29E_0084	Configuration (FTM2_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
29E_0088	FTM Fault Input Polarity (FTM2_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
29E_008C	Synchronization Configuration (FTM2_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
29E_0090	FTM Inverting Control (FTM2_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
29E_0094	FTM Software Output Control (FTM2_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
29E_0098	FTM PWM Load (FTM2_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
29F_0000	Status And Control (FTM3_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
29F_0004	Counter (FTM3_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
29F_0008	Modulo (FTM3_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
29F_000C	Channel (n) Status And Control (FTM3_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0010	Channel (n) Value (FTM3_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_0014	Channel (n) Status And Control (FTM3_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0018	Channel (n) Value (FTM3_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_001C	Channel (n) Status And Control (FTM3_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0020	Channel (n) Value (FTM3_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_0024	Channel (n) Status And Control (FTM3_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0028	Channel (n) Value (FTM3_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_002C	Channel (n) Status And Control (FTM3_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0030	Channel (n) Value (FTM3_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_0034	Channel (n) Status And Control (FTM3_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0038	Channel (n) Value (FTM3_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_003C	Channel (n) Status And Control (FTM3_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0040	Channel (n) Value (FTM3_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_0044	Channel (n) Status And Control (FTM3_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
29F_0048	Channel (n) Value (FTM3_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
29F_004C	Counter Initial Value (FTM3_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
29F_0050	Capture And Compare Status (FTM3_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
29F_0054	Features Mode Selection (FTM3_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
29F_0058	Synchronization (FTM3_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
29F_005C	Initial State For Channels Output (FTM3_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
29F_0060	Output Mask (FTM3_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
29F_0064	Function For Linked Channels (FTM3_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>

Table continues on the next page...

**FTM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
29F_0068	Deadtime Insertion Control (FTM3_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
29F_006C	FTM External Trigger (FTM3_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
29F_0070	Channels Polarity (FTM3_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
29F_0074	Fault Mode Status (FTM3_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
29F_0078	Input Capture Filter Control (FTM3_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
29F_007C	Fault Control (FTM3_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
29F_0080	Quadrature Decoder Control And Status (FTM3_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
29F_0084	Configuration (FTM3_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
29F_0088	FTM Fault Input Polarity (FTM3_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
29F_008C	Synchronization Configuration (FTM3_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
29F_0090	FTM Inverting Control (FTM3_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
29F_0094	FTM Software Output Control (FTM3_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
29F_0098	FTM PWM Load (FTM3_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
2A0_0000	Status And Control (FTM4_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
2A0_0004	Counter (FTM4_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
2A0_0008	Modulo (FTM4_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
2A0_000C	Channel (n) Status And Control (FTM4_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0010	Channel (n) Value (FTM4_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_0014	Channel (n) Status And Control (FTM4_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0018	Channel (n) Value (FTM4_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_001C	Channel (n) Status And Control (FTM4_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0020	Channel (n) Value (FTM4_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_0024	Channel (n) Status And Control (FTM4_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0028	Channel (n) Value (FTM4_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_002C	Channel (n) Status And Control (FTM4_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0030	Channel (n) Value (FTM4_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_0034	Channel (n) Status And Control (FTM4_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0038	Channel (n) Value (FTM4_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_003C	Channel (n) Status And Control (FTM4_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2A0_0040	Channel (n) Value (FTM4_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_0044	Channel (n) Status And Control (FTM4_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A0_0048	Channel (n) Value (FTM4_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A0_004C	Counter Initial Value (FTM4_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
2A0_0050	Capture And Compare Status (FTM4_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
2A0_0054	Features Mode Selection (FTM4_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
2A0_0058	Synchronization (FTM4_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
2A0_005C	Initial State For Channels Output (FTM4_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
2A0_0060	Output Mask (FTM4_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
2A0_0064	Function For Linked Channels (FTM4_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
2A0_0068	Deadtime Insertion Control (FTM4_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
2A0_006C	FTM External Trigger (FTM4_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
2A0_0070	Channels Polarity (FTM4_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
2A0_0074	Fault Mode Status (FTM4_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
2A0_0078	Input Capture Filter Control (FTM4_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
2A0_007C	Fault Control (FTM4_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
2A0_0080	Quadrature Decoder Control And Status (FTM4_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
2A0_0084	Configuration (FTM4_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
2A0_0088	FTM Fault Input Polarity (FTM4_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
2A0_008C	Synchronization Configuration (FTM4_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
2A0_0090	FTM Inverting Control (FTM4_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
2A0_0094	FTM Software Output Control (FTM4_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
2A0_0098	FTM PWM Load (FTM4_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
2A1_0000	Status And Control (FTM5_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
2A1_0004	Counter (FTM5_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2A1_0008	Modulo (FTM5_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
2A1_000C	Channel (n) Status And Control (FTM5_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0010	Channel (n) Value (FTM5_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_0014	Channel (n) Status And Control (FTM5_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0018	Channel (n) Value (FTM5_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_001C	Channel (n) Status And Control (FTM5_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0020	Channel (n) Value (FTM5_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_0024	Channel (n) Status And Control (FTM5_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0028	Channel (n) Value (FTM5_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_002C	Channel (n) Status And Control (FTM5_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0030	Channel (n) Value (FTM5_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_0034	Channel (n) Status And Control (FTM5_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0038	Channel (n) Value (FTM5_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_003C	Channel (n) Status And Control (FTM5_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0040	Channel (n) Value (FTM5_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_0044	Channel (n) Status And Control (FTM5_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A1_0048	Channel (n) Value (FTM5_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A1_004C	Counter Initial Value (FTM5_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
2A1_0050	Capture And Compare Status (FTM5_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
2A1_0054	Features Mode Selection (FTM5_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
2A1_0058	Synchronization (FTM5_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
2A1_005C	Initial State For Channels Output (FTM5_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
2A1_0060	Output Mask (FTM5_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
2A1_0064	Function For Linked Channels (FTM5_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
2A1_0068	Deadtime Insertion Control (FTM5_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
2A1_006C	FTM External Trigger (FTM5_EXTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
2A1_0070	Channels Polarity (FTM5_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
2A1_0074	Fault Mode Status (FTM5_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
2A1_0078	Input Capture Filter Control (FTM5_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
2A1_007C	Fault Control (FTM5_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2A1_0080	Quadrature Decoder Control And Status (FTM5_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
2A1_0084	Configuration (FTM5_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
2A1_0088	FTM Fault Input Polarity (FTM5_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
2A1_008C	Synchronization Configuration (FTM5_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
2A1_0090	FTM Inverting Control (FTM5_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
2A1_0094	FTM Software Output Control (FTM5_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
2A1_0098	FTM PWM Load (FTM5_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
2A2_0000	Status And Control (FTM6_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
2A2_0004	Counter (FTM6_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
2A2_0008	Modulo (FTM6_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
2A2_000C	Channel (n) Status And Control (FTM6_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0010	Channel (n) Value (FTM6_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_0014	Channel (n) Status And Control (FTM6_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0018	Channel (n) Value (FTM6_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_001C	Channel (n) Status And Control (FTM6_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0020	Channel (n) Value (FTM6_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_0024	Channel (n) Status And Control (FTM6_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0028	Channel (n) Value (FTM6_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_002C	Channel (n) Status And Control (FTM6_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0030	Channel (n) Value (FTM6_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_0034	Channel (n) Status And Control (FTM6_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0038	Channel (n) Value (FTM6_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_003C	Channel (n) Status And Control (FTM6_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0040	Channel (n) Value (FTM6_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_0044	Channel (n) Status And Control (FTM6_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A2_0048	Channel (n) Value (FTM6_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A2_004C	Counter Initial Value (FTM6_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
2A2_0050	Capture And Compare Status (FTM6_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
2A2_0054	Features Mode Selection (FTM6_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
2A2_0058	Synchronization (FTM6_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
2A2_005C	Initial State For Channels Output (FTM6_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2A2_0060	Output Mask (FTM6_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
2A2_0064	Function For Linked Channels (FTM6_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
2A2_0068	Deadtime Insertion Control (FTM6_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
2A2_006C	FTM External Trigger (FTM6_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
2A2_0070	Channels Polarity (FTM6_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
2A2_0074	Fault Mode Status (FTM6_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
2A2_0078	Input Capture Filter Control (FTM6_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
2A2_007C	Fault Control (FTM6_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
2A2_0080	Quadrature Decoder Control And Status (FTM6_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
2A2_0084	Configuration (FTM6_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
2A2_0088	FTM Fault Input Polarity (FTM6_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
2A2_008C	Synchronization Configuration (FTM6_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
2A2_0090	FTM Inverting Control (FTM6_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
2A2_0094	FTM Software Output Control (FTM6_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>
2A2_0098	FTM PWM Load (FTM6_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
2A3_0000	Status And Control (FTM7_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
2A3_0004	Counter (FTM7_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
2A3_0008	Modulo (FTM7_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
2A3_000C	Channel (n) Status And Control (FTM7_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0010	Channel (n) Value (FTM7_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_0014	Channel (n) Status And Control (FTM7_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0018	Channel (n) Value (FTM7_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_001C	Channel (n) Status And Control (FTM7_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0020	Channel (n) Value (FTM7_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_0024	Channel (n) Status And Control (FTM7_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0028	Channel (n) Value (FTM7_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_002C	Channel (n) Status And Control (FTM7_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2A3_0030	Channel (n) Value (FTM7_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_0034	Channel (n) Status And Control (FTM7_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0038	Channel (n) Value (FTM7_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_003C	Channel (n) Status And Control (FTM7_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0040	Channel (n) Value (FTM7_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_0044	Channel (n) Status And Control (FTM7_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A3_0048	Channel (n) Value (FTM7_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A3_004C	Counter Initial Value (FTM7_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
2A3_0050	Capture And Compare Status (FTM7_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
2A3_0054	Features Mode Selection (FTM7_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
2A3_0058	Synchronization (FTM7_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
2A3_005C	Initial State For Channels Output (FTM7_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
2A3_0060	Output Mask (FTM7_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
2A3_0064	Function For Linked Channels (FTM7_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
2A3_0068	Deadtime Insertion Control (FTM7_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
2A3_006C	FTM External Trigger (FTM7_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
2A3_0070	Channels Polarity (FTM7_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
2A3_0074	Fault Mode Status (FTM7_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>
2A3_0078	Input Capture Filter Control (FTM7_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/1089</a>
2A3_007C	Fault Control (FTM7_FLCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/1090</a>
2A3_0080	Quadrature Decoder Control And Status (FTM7_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/1092</a>
2A3_0084	Configuration (FTM7_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/1094</a>
2A3_0088	FTM Fault Input Polarity (FTM7_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/1095</a>
2A3_008C	Synchronization Configuration (FTM7_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/1097</a>
2A3_0090	FTM Inverting Control (FTM7_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/1099</a>
2A3_0094	FTM Software Output Control (FTM7_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/1100</a>

Table continues on the next page...

## FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2A3_0098	FTM PWM Load (FTM7_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/1102</a>
2A4_0000	Status And Control (FTM8_SC)	32	R/W	0000_0000h	<a href="#">23.4.3/1061</a>
2A4_0004	Counter (FTM8_CNT)	32	R/W	0000_0000h	<a href="#">23.4.4/1062</a>
2A4_0008	Modulo (FTM8_MOD)	32	R/W	0000_0000h	<a href="#">23.4.5/1063</a>
2A4_000C	Channel (n) Status And Control (FTM8_C0SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0010	Channel (n) Value (FTM8_C0V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_0014	Channel (n) Status And Control (FTM8_C1SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0018	Channel (n) Value (FTM8_C1V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_001C	Channel (n) Status And Control (FTM8_C2SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0020	Channel (n) Value (FTM8_C2V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_0024	Channel (n) Status And Control (FTM8_C3SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0028	Channel (n) Value (FTM8_C3V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_002C	Channel (n) Status And Control (FTM8_C4SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0030	Channel (n) Value (FTM8_C4V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_0034	Channel (n) Status And Control (FTM8_C5SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0038	Channel (n) Value (FTM8_C5V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_003C	Channel (n) Status And Control (FTM8_C6SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0040	Channel (n) Value (FTM8_C6V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_0044	Channel (n) Status And Control (FTM8_C7SC)	32	R/W	0000_0000h	<a href="#">23.4.6/1064</a>
2A4_0048	Channel (n) Value (FTM8_C7V)	32	R/W	0000_0000h	<a href="#">23.4.7/1066</a>
2A4_004C	Counter Initial Value (FTM8_CNTIN)	32	R/W	0000_0000h	<a href="#">23.4.8/1067</a>
2A4_0050	Capture And Compare Status (FTM8_STATUS)	32	R/W	0000_0000h	<a href="#">23.4.9/1067</a>
2A4_0054	Features Mode Selection (FTM8_MODE)	32	R/W	0000_0004h	<a href="#">23.4.10/1069</a>
2A4_0058	Synchronization (FTM8_SYNC)	32	R/W	0000_0000h	<a href="#">23.4.11/1071</a>
2A4_005C	Initial State For Channels Output (FTM8_OUTINIT)	32	R/W	0000_0000h	<a href="#">23.4.12/1074</a>
2A4_0060	Output Mask (FTM8_OUTMASK)	32	R/W	0000_0000h	<a href="#">23.4.13/1075</a>
2A4_0064	Function For Linked Channels (FTM8_COMBINE)	32	R/W	0000_0000h	<a href="#">23.4.14/1077</a>
2A4_0068	Deadtime Insertion Control (FTM8_DEADTIME)	32	R/W	0000_0000h	<a href="#">23.4.15/1082</a>
2A4_006C	FTM External Trigger (FTM8_EXTTRIG)	32	R/W	0000_0000h	<a href="#">23.4.16/1083</a>
2A4_0070	Channels Polarity (FTM8_POL)	32	R/W	0000_0000h	<a href="#">23.4.17/1085</a>
2A4_0074	Fault Mode Status (FTM8_FMS)	32	R/W	0000_0000h	<a href="#">23.4.18/1087</a>

Table continues on the next page...

**FTM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2A4_0078	Input Capture Filter Control (FTM8_FILTER)	32	R/W	0000_0000h	<a href="#">23.4.19/ 1089</a>
2A4_007C	Fault Control (FTM8_FLTCTRL)	32	R/W	0000_0000h	<a href="#">23.4.20/ 1090</a>
2A4_0080	Quadrature Decoder Control And Status (FTM8_QDCTRL)	32	R/W	0000_0000h	<a href="#">23.4.21/ 1092</a>
2A4_0084	Configuration (FTM8_CONF)	32	R/W	0000_0000h	<a href="#">23.4.22/ 1094</a>
2A4_0088	FTM Fault Input Polarity (FTM8_FLTPOL)	32	R/W	0000_0000h	<a href="#">23.4.23/ 1095</a>
2A4_008C	Synchronization Configuration (FTM8_SYNCONF)	32	R/W	0000_0000h	<a href="#">23.4.24/ 1097</a>
2A4_0090	FTM Inverting Control (FTM8_INVCTRL)	32	R/W	0000_0000h	<a href="#">23.4.25/ 1099</a>
2A4_0094	FTM Software Output Control (FTM8_SWOCTRL)	32	R/W	0000_0000h	<a href="#">23.4.26/ 1100</a>
2A4_0098	FTM PWM Load (FTM8_PWMLOAD)	32	R/W	0000_0000h	<a href="#">23.4.27/ 1102</a>

### 23.4.3 Status And Control (FTMx\_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, and prescaler factor. These controls relate to all channels within this module.

Address: Base address + 0h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									TOF		TOIE		CPWMS		CLKS	PS
W									0		0		0		0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_SC field descriptions**

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 TOF	Timer Overflow Flag  Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect.  If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case, a TOF interrupt request is not lost due to the clearing sequence for a previous TOF.  0 FTM counter has not overflowed. 1 FTM counter has overflowed.

*Table continues on the next page...*

**FTMx\_SC field descriptions (continued)**

Field	Description
25 TOIE	<p>Timer Overflow Interrupt Enable</p> <p>Enables FTM overflow interrupts.</p> <p>0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.</p>
26 CPWMS	<p>Center-Aligned PWM Select</p> <p>Selects CPWM mode. This mode configures the FTM to operate in Up-Down Counting mode.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 FTM counter operates in Up Counting mode. 1 FTM counter operates in Up-Down Counting mode.</p>
27–28 CLKS	<p>Clock Source Selection</p> <p>Selects FTM counter clock sources.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>00 No clock selected. This in effect disables the FTM counter. 01 System clock 10 Fixed frequency clock 11 External clock</p>
29–31 PS	<p>Prescale Factor Selection</p> <p>Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128</p>

**23.4.4 Counter (FTMx\_CNT)**

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

When is active, the FTM counter is frozen. This is the value that you may read.

Address: Base address + 4h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R									0																								
W																																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### FTMx\_CNT field descriptions

Field	Description
0–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16–31 COUNT	Counter Value

### 23.4.5 Modulo (FTMx\_MOD)

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock, and the next value of FTM counter depends on the selected counting method; see [Counter](#).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to [Registers updated from write buffers](#).

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the SC register whether is active or not.

Initialize the FTM counter, by writing to CNT, before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Address: Base address + 8h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																																	
W																																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### FTMx\_MOD field descriptions

Field	Description
0–15 Reserved	This field is reserved.
16–31 MOD	Modulo Value

### 23.4.6 Channel (n) Status And Control (FTMx\_CnSC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

**Table 23-6. Mode, edge, and level selection**

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	X	X	XX	00	Pin not used for FTM—revert the channel pin to general purpose I/O or other peripheral control	
0	0	0	00	01	Input Capture	Capture on Rising Edge Only
				10		Capture on Falling Edge Only
				11		Capture on Rising or Falling Edge
				01	Output Compare	Toggle Output on match
				10		Clear Output on match
				11		Set Output on match
			1X	10	Edge-Aligned PWM	High-true pulses (clear Output on match)
				X1		Low-true pulses (set Output on match)
			1	10	Center-Aligned PWM	High-true pulses (clear Output on match-up)
				X1		Low-true pulses (set Output on match-up)
			0	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
				X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)

Table continues on the next page...

**Table 23-6. Mode, edge, and level selection (continued)**

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
1	0	0	X0	See the following table (Table 23-7).	Dual Edge Capture	One-Shot Capture mode
			X1			Continuous Capture mode

**Table 23-7. Dual Edge Capture mode — edge polarity selection**

ELSnB	ELSnA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

Address: Base address + Ch offset + (8d × i), where i=0d to 7d

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R										0							
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R										CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
W										0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**FTMx\_CnSC field descriptions**

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 CHF	Channel Flag  Set by hardware when an event occurs on the channel. CHF is cleared by reading the CSC register while CHnF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect.  If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.  0 No channel event has occurred. 1 A channel event has occurred.
25 CHIE	Channel Interrupt Enable  Enables channel interrupts.  0 Disable channel interrupts. Use software polling. 1 Enable channel interrupts.

Table continues on the next page...

**FTMx\_CnSC field descriptions (continued)**

Field	Description
26 MSB	Channel Mode Select  Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See <a href="#">Table 23-6</a> .  This field is write protected. It can be written only when MODE[WPDIS] = 1.
27 MSA	Channel Mode Select  Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See <a href="#">Table 23-6</a> .  This field is write protected. It can be written only when MODE[WPDIS] = 1.
28 ELSB	Edge or Level Select  The functionality of ELSB and ELSA depends on the channel mode. See <a href="#">Table 23-6</a> .  This field is write protected. It can be written only when MODE[WPDIS] = 1.
29 ELSA	Edge or Level Select  The functionality of ELSB and ELSA depends on the channel mode. See <a href="#">Table 23-6</a> .  This field is write protected. It can be written only when MODE[WPDIS] = 1.
30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
31 DMA	DMA Enable  Enables DMA transfers for the channel.  0 Disable DMA transfers. 1 Enable DMA transfers.

**23.4.7 Channel (n) Value (FTMx\_CnV)**

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In Input Capture, Capture Test, and Dual Edge Capture modes, any write to a CnV register is ignored.

In output modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to [Registers updated from write buffers](#).

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the CnSC register whether mode is active or not.

Address: Base address + 10h offset + (8d × i), where i=0d to 7d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### FTMx\_CnV field descriptions

Field	Description
0–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16–31 VAL	Channel Value  Captured FTM counter value of the input modes or the match value for the output modes

#### 23.4.8 Counter Initial Value (FTMx\_CNTIN)

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to [Registers updated from write buffers](#).

When the FTM clock is initially selected, by writing a non-zero value to the CLKS bits, the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the CNTIN register and then initialize the FTM counter by writing any value to the CNT register.

Address: Base address + 4Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### FTMx\_CNTIN field descriptions

Field	Description
0–15 Reserved	This field is reserved.
16–31 INIT	Initial Value Of The FTM Counter

#### 23.4.9 Capture And Compare Status (FTMx\_STATUS)

The STATUS register contains a copy of the status flag CHnF bit in CnSC for each FTM channel for software convenience.

## Memory map and register definition

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHnF is cleared by reading STATUS while CHnF is set and then writing a 0 to the CHnF bit. Writing a 1 to CHnF has no effect.

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHnF remains set indicating an event has occurred. In this case, a CHnF interrupt request is not lost due to the clearing sequence for a previous CHnF.

Address: Base address + 50h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## FTMx\_STATUS field descriptions

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 CH7F	Channel 7 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
25 CH6F	Channel 6 Flag

*Table continues on the next page...*

**FTMx\_STATUS field descriptions (continued)**

Field	Description
	See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
26 CH5F	Channel 5 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
27 CH4F	Channel 4 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
28 CH3F	Channel 3 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
29 CH2F	Channel 2 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
30 CH1F	Channel 1 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.
31 CH0F	Channel 0 Flag  See the register description.  0 No channel event has occurred. 1 A channel event has occurred.

**23.4.10 Features Mode Selection (FTMx\_MODE)**

This register contains the global enable bit for FTM-specific features and the control bits used to configure:

- Fault control mode and interrupt
- Capture Test mode
- PWM synchronization

## Memory map and register definition

- Write protection
- Channel output initialization

These controls relate to all channels within this module.

Address: Base address + 54h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0								FAULTIE	FAULTM		CAPTEST	PWM_SYNC	WPDIS	INIT	FTMEN
W									0	0	0	0	0	1	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### FTMx\_MODE field descriptions

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 FAULTIE	Fault Interrupt Enable  Enables the generation of an interrupt when a fault is detected by FTM and the FTM fault control is enabled.  0 Fault control interrupt is disabled. 1 Fault control interrupt is enabled.
25–26 FAULTM	Fault Control Mode  Defines the FTM fault control mode.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  00 Fault control is disabled for all channels. 01 Fault control is enabled for even channels only (channels 0, 2, 4, and 6), and the selected mode is the manual fault clearing. 10 Fault control is enabled for all channels, and the selected mode is the manual fault clearing. 11 Fault control is enabled for all channels, and the selected mode is the automatic fault clearing.
27 CAPTEST	Capture Test Mode Enable  Enables the capture test mode.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Capture test mode is disabled. 1 Capture test mode is enabled.
28 PWM_SYNC	PWM Synchronization Mode

Table continues on the next page...

**FTMx\_MODE field descriptions (continued)**

Field	Description
	Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. See <a href="#">PWM synchronization</a> . The PWMSYNC bit configures the synchronization when SYNCMODE is 0.
	<ul style="list-style-type: none"> <li>0 No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization.</li> <li>1 Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.</li> </ul>
29 WPDIS	<p>Write Protection Disable</p> <p>When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect.</p> <ul style="list-style-type: none"> <li>0 Write protection is enabled.</li> <li>1 Write protection is disabled.</li> </ul>
30 INIT	<p>Initialize The Channels Output</p> <p>When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect.</p> <p>The INIT bit is always read as 0.</p>
31 FTMEN	<p>FTM Enable</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <ul style="list-style-type: none"> <li>0 TPM compatibility. Free running counter and synchronization compatible with TPM.</li> <li>1 Free running counter and synchronization are different from TPM behavior.</li> </ul>

### 23.4.11 Synchronization (FTMx\_SYNC)

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

**NOTE**

The software trigger, SWSYNC bit, and hardware triggers TRIG0, TRIG1, and TRIG2 bits have a potential conflict if used together when SYNCMODE = 0. Use only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.

## Memory map and register definition

The selection of the loading point, CNTMAX and CNTMIN bits, is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection, TRIG0, TRIG1, or TRIG2 bits, is likely to result in unpredictable behavior.

The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See [PWM synchronization](#).

Address: Base address + 58h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									SW/SYNC	TRIG2	TRIG1	TRIG0	SYNCHOM	REINIT	CNTMAX	CNTMIN
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## FTMx\_SYNC field descriptions

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 SWSYNC	PWM Synchronization Software Trigger  Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit.  0 Software trigger is not selected. 1 Software trigger is selected.
25 TRIG2	PWM Synchronization Hardware Trigger 2  Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal.  0 Trigger is disabled. 1 Trigger is enabled.
26 TRIG1	PWM Synchronization Hardware Trigger 1  Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal.

Table continues on the next page...

**FTMx\_SYNC field descriptions (continued)**

Field	Description
	<p>0 Trigger is disabled. 1 Trigger is enabled.</p>
27 TRIG0	<p>PWM Synchronization Hardware Trigger 0  Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 occurs when a rising edge is detected at the trigger 0 input signal.</p> <p>0 Trigger is disabled. 1 Trigger is enabled.</p>
28 SYNCOM	<p>Output Mask Synchronization  Selects when the OUTMASK register is updated with the value of its buffer.</p> <p>0 OUTMASK register is updated with the value of its buffer in all rising edges of the system clock. 1 OUTMASK register is updated with the value of its buffer only by the PWM synchronization.</p>
29 REINIT	<p>FTM Counter Reinitialization By Synchronization (<a href="#">FTM counter synchronization</a>)  Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected. The REINIT bit configures the synchronization when SYNCMODE is zero.</p> <p>0 FTM counter continues to count normally. 1 FTM counter is updated with its initial value when the selected trigger is detected.</p>
30 CNTMAX	<p>Maximum Loading Point Enable  Selects the maximum loading point to PWM synchronization. See <a href="#">Boundary cycle and loading points</a>. If CNTMAX is 1, the selected loading point is when the FTM counter reaches its maximum value (MOD register).</p> <p>0 The maximum loading point is disabled. 1 The maximum loading point is enabled.</p>
31 CNTMIN	<p>Minimum Loading Point Enable  Selects the minimum loading point to PWM synchronization. See <a href="#">Boundary cycle and loading points</a>. If CNTMIN is one, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register).</p> <p>0 The minimum loading point is disabled. 1 The minimum loading point is enabled.</p>

## 23.4.12 Initial State For Channels Output (FTMx\_OUTINIT)

Address: Base address + 5Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CH7OI	CH6OI	CH5OI	CH4OI	CH3OI	CH2OI	CH1OI	CH0OI
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### FTMx\_OUTINIT field descriptions

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 CH7OI	Channel 7 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
25 CH6OI	Channel 6 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
26 CH5OI	Channel 5 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
27 CH4OI	Channel 4 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
28 CH3OI	Channel 3 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.

Table continues on the next page...

**FTMx\_OUTINIT field descriptions (continued)**

Field	Description
	0 The initialization value is 0. 1 The initialization value is 1.
29 CH2OI	Channel 2 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
30 CH1OI	Channel 1 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.
31 CH0OI	Channel 0 Output Initialization Value  Selects the value that is forced into the channel output when the initialization occurs.  0 The initialization value is 0. 1 The initialization value is 1.

**23.4.13 Output Mask (FTMx\_OUTMASK)**

This register provides a mask for each FTM channel. The mask of a channel determines if its output responds, that is, it is masked or not, when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to [PWM synchronization](#).

Address: Base address + 60h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CH7OM	CH6OM	CH5OM	CH4OM	CH3OM	CH2OM	CH1OM	CH0OM
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_OUTMASK field descriptions**

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 CH7OM	Channel 7 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
25 CH6OM	Channel 6 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
26 CH5OM	Channel 5 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
27 CH4OM	Channel 4 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
28 CH3OM	Channel 3 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
29 CH2OM	Channel 2 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
30 CH1OM	Channel 1 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
31 CH0OM	Channel 0 Output Mask  Defines if the channel output is masked or unmasked.  0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.

### 23.4.14 Function For Linked Channels (FTMx\_COMBINE)

This register contains the control bits used to configure the fault control, synchronization, deadtime insertion, Dual Edge Capture mode, Complementary, and Combine mode for each pair of channels (n) and (n+1), where n equals 0, 2, 4, and 6.

Address: Base address + 64h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	0	FAULTEN3	SYNCEN3	DTEN3	DECAP3	DECAPEN3	COMP3	COMBINE3	0	FAULTEN2	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	FAULTEN1	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	0	FAULTENO	SYNCENO	DTEN0	DECAP0	DECAPENO	COMP0	COMBINE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### FTMx\_COMBINE field descriptions

Field	Description
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 FAULTEN3	Fault Control Enable For n = 6 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
2 SYNCEN3	Synchronization Enable For n = 6 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
3 DTEN3	Deadtime Enable For n = 6 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.

*Table continues on the next page...*

## FTMx\_COMBINE field descriptions (continued)

Field	Description
	<p>0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.</p>
4 DECAP3	<p>Dual Edge Capture Mode Captures For n = 6 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.</p> <p>0 The dual edge captures are inactive. 1 The dual edge captures are active.</p>
5 DECAPEN3	<p>Dual Edge Capture Mode Enable For n = 6 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to <a href="#">Table 23-6</a>. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.</p>
6 COMP3	<p>Complement Of Channel (n) for n = 6 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.</p>
7 COMBINE3	<p>Combine Channels For n = 6 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.</p>
8 Reserved	<p>This field is reserved. This read-only field is reserved and always has the value 0.</p>
9 FAULTEN2	<p>Fault Control Enable For n = 4 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.</p>
10 SYNCEN2	<p>Synchronization Enable For n = 4 Enables PWM synchronization of registers C(n)V and C(n+1)V.</p>

*Table continues on the next page...*

**FTMx\_COMBINE field descriptions (continued)**

Field	Description
	<p>0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.</p>
11 DTEN2	<p>Deadtime Enable For n = 4 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.</p>
12 DECAP2	<p>Dual Edge Capture Mode Captures For n = 4 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.</p> <p>0 The dual edge captures are inactive. 1 The dual edge captures are active.</p>
13 DECAPEN2	<p>Dual Edge Capture Mode Enable For n = 4 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to <a href="#">Table 23-6</a>. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.</p>
14 COMP2	<p>Complement Of Channel (n) For n = 4 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.</p>
15 COMBINE2	<p>Combine Channels For n = 4 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.</p>
16 Reserved	<p>This field is reserved. This read-only field is reserved and always has the value 0.</p>
17 FAULTEN1	<p>Fault Control Enable For n = 2 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p>

*Table continues on the next page...*

## FTMx\_COMBINE field descriptions (continued)

Field	Description
	<p>0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.</p>
18 SYNCEN1	<p>Synchronization Enable For n = 2 Enables PWM synchronization of registers C(n)V and C(n+1)V.</p> <p>0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.</p>
19 DTEN1	<p>Deadtime Enable For n = 2 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.</p>
20 DECAP1	<p>Dual Edge Capture Mode Captures For n = 2 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if Dual Edge Capture – One-Shot mode is selected and when the capture of channel (n+1) event is made.</p> <p>0 The dual edge captures are inactive. 1 The dual edge captures are active.</p>
21 DECAPEN1	<p>Dual Edge Capture Mode Enable For n = 2 Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to <a href="#">Table 23-6</a>. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.</p>
22 COMP1	<p>Complement Of Channel (n) For n = 2 Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.</p>
23 COMBINE1	<p>Combine Channels For n = 2 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.</p>

Table continues on the next page...

**FTMx\_COMBINE field descriptions (continued)**

Field	Description
24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 FAULTEN0	Fault Control Enable For n = 0  Enables the fault control in channels (n) and (n+1).  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
26 SYNCEN0	Synchronization Enable For n = 0  Enables PWM synchronization of registers C(n)V and C(n+1)V.  0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
27 DTEN0	Deadtime Enable For n = 0  Enables the deadtime insertion in the channels (n) and (n+1).  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
28 DECAP0	Dual Edge Capture Mode Captures For n = 0  Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits.  This field applies only when DECAPEN = 1.  DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.  0 The dual edge captures are inactive. 1 The dual edge captures are active.
29 DECAPEN0	Dual Edge Capture Mode Enable For n = 0  Enables the Dual Edge Capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in Dual Edge Capture mode according to <a href="#">Table 23-6</a> .  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 The Dual Edge Capture mode in this pair of channels is disabled. 1 The Dual Edge Capture mode in this pair of channels is enabled.
30 COMP0	Complement Of Channel (n) For n = 0  Enables Complementary mode for the combined channels. In Complementary mode the channel (n+1) output is the inverse of the channel (n) output.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.

*Table continues on the next page...*

**FTMx\_COMBINE field descriptions (continued)**

Field	Description
31 COMBINE0	<p>Combine Channels For n = 0</p> <p>Enables the combine feature for channels (n) and (n+1).</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.</p>

**23.4.15 Deadtime Insertion Control (FTMx\_DEADTIME)**

This register selects the deadtime prescaler factor and deadtime value. All FTM channels use this clock prescaler and this deadtime value for the deadtime insertion.

Address: Base address + 68h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	0								DTPS			DTVAL				
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**FTMx\_DEADTIME field descriptions**

Field	Description
0–23 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
24–25 DTPS	<p>Deadtime Prescaler Value</p> <p>Selects the division factor of the system clock. This prescaled clock is used by the deadtime counter.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0x Divide the system clock by 1. 10 Divide the system clock by 4. 11 Divide the system clock by 16.</p>
26–31 DTVAL	<p>Deadtime Value</p> <p>Selects the deadtime insertion value for the deadtime counter. The deadtime counter is clocked by a scaled version of the system clock. See the description of DTPS.</p> <p>Deadtime insert value = (DTPS × DTVAL).</p> <p>DTVAL selects the number of deadtime counts inserted as follows:</p> <p>When DTVAL is 0, no counts are inserted.</p> <p>When DTVAL is 1, 1 count is inserted.</p> <p>When DTVAL is 2, 2 counts are inserted.</p> <p>This pattern continues up to a possible 63 counts.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p>

### 23.4.16 FTM External Trigger (FTMx\_EXTTRIG)

This register:

- Indicates when a channel trigger was generated
- Enables the generation of a trigger when the FTM counter is equal to its initial value
- Selects which channels are used in the generation of the channel triggers

Several channels can be selected to generate multiple triggers in one PWM period. See [Channel trigger output](#) and [Initialization trigger](#).

Channels 6 and 7 are not used to generate channel triggers.

Address: Base address + 6Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									TRIGF		INITTRIGEN		CH1TRIG		CH5TRIG	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_EXTTRIG field descriptions**

Field	Description
0–23 Reserved	This field is reserved.
24 TRIGF	<p>Channel Trigger Flag</p> <p>Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect.</p> <p>If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF.</p> <p>0 No channel trigger was generated. 1 A channel trigger was generated.</p>
25 INITTRIGEN	<p>Initialization Trigger Enable</p> <p>Enables the generation of the trigger when the FTM counter is equal to the CNTIN register.</p> <p>0 The generation of initialization trigger is disabled. 1 The generation of initialization trigger is enabled.</p>
26 CH1TRIG	<p>Channel 1 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p> <p>0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.</p>
27 CH0TRIG	<p>Channel 0 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p> <p>0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.</p>
28 CH5TRIG	<p>Channel 5 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p> <p>0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.</p>
29 CH4TRIG	<p>Channel 4 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p> <p>0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.</p>
30 CH3TRIG	<p>Channel 3 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p> <p>0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.</p>
31 CH2TRIG	<p>Channel 2 Trigger Enable</p> <p>Enables the generation of the channel trigger when the FTM counter is equal to the CnV register.</p>

*Table continues on the next page...*

**FTMx\_EXTTRIG field descriptions (continued)**

Field	Description
	0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.

**23.4.17 Channels Polarity (FTMx\_POL)**

This register defines the output polarity of the FTM channels.

**NOTE**

The safe value that is driven in a channel output when the fault control is enabled and a fault condition is detected is the inactive state of the channel. That is, the safe value of a channel is the value of its POL bit.

Address: Base address + 70h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved								POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_POL field descriptions**

Field	Description
0–23 Reserved	This field is reserved.
24 POL7	Channel 7 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
25 POL6	Channel 6 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
26 POL5	Channel 5 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.

*Table continues on the next page...*

**FTMx\_POL field descriptions (continued)**

Field	Description
	<p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>
27 POL4	<p>Channel 4 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>
28 POL3	<p>Channel 3 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>
29 POL2	<p>Channel 2 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>
30 POL1	<p>Channel 1 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>
31 POL0	<p>Channel 0 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The channel polarity is active high. 1 The channel polarity is active low.</p>

### 23.4.18 Fault Mode Status (FTMx\_FMS)

This register contains the fault detection flags, write protection enable bit, and the logic OR of the enabled fault inputs.

Address: Base address + 74h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0								FAULTF	WPEN	FAULTIN	0	FAULTF3	FAULTF2	FAULTF1	FAULTF0
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### FTMx\_FMS field descriptions

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 FAULTF	Fault Detection Flag  Represents the logic OR of the individual FAULTF $j$ bits where $j = 3, 2, 1, 0$ . Clear FAULTF by reading the FMS register while FAULTF is set and then writing a 0 to FAULTF while there is no existing fault condition at the enabled fault inputs. Writing a 1 to FAULTF has no effect.  If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTF remains set after the clearing sequence is completed for the earlier fault condition. FAULTF is also cleared when FAULTF $j$ bits are cleared individually.  0 No fault condition was detected. 1 A fault condition was detected.

Table continues on the next page...

**FTMx\_FMS field descriptions (continued)**

Field	Description
25 WPEN	<p>Write Protection Enable</p> <p>The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect.</p> <p>0 Write protection is disabled. Write protected bits can be written. 1 Write protection is enabled. Write protected bits cannot be written.</p>
26 FAULTIN	<p>Fault Inputs</p> <p>Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled.</p> <p>0 The logic OR of the enabled fault inputs is 0. 1 The logic OR of the enabled fault inputs is 1.</p>
27 Reserved	<p>This field is reserved. This read-only field is reserved and always has the value 0.</p>
28 FAULTF3	<p>Fault Detection Flag 3</p> <p>Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.</p> <p>Clear FAULTF3 by reading the FMS register while FAULTF3 is set and then writing a 0 to FAULTF3 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF3 has no effect. FAULTF3 bit is also cleared when FAULTF bit is cleared.</p> <p>If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF3 remains set after the clearing sequence is completed for the earlier fault condition.</p> <p>0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.</p>
29 FAULTF2	<p>Fault Detection Flag 2</p> <p>Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.</p> <p>Clear FAULTF2 by reading the FMS register while FAULTF2 is set and then writing a 0 to FAULTF2 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF2 has no effect. FAULTF2 bit is also cleared when FAULTF bit is cleared.</p> <p>If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF2 remains set after the clearing sequence is completed for the earlier fault condition.</p> <p>0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.</p>
30 FAULTF1	<p>Fault Detection Flag 1</p> <p>Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.</p> <p>Clear FAULTF1 by reading the FMS register while FAULTF1 is set and then writing a 0 to FAULTF1 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF1 has no effect. FAULTF1 bit is also cleared when FAULTF bit is cleared.</p>

*Table continues on the next page...*

**FTMx\_FMS field descriptions (continued)**

Field	Description
	If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF1 remains set after the clearing sequence is completed for the earlier fault condition.  0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
31 FAULTF0	Fault Detection Flag 0  Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input.  Clear FAULTF0 by reading the FMS register while FAULTF0 is set and then writing a 0 to FAULTF0 while there is no existing fault condition at the corresponding fault input. Writing a 1 to FAULTF0 has no effect. FAULTF0 bit is also cleared when FAULTF bit is cleared.  If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF0 remains set after the clearing sequence is completed for the earlier fault condition.  0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.

**23.4.19 Input Capture Filter Control (FTMx\_FILTER)**

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

**NOTE**

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

Address: Base address + 78h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	Reserved				CH3FVAL	CH2FVAL	CH1FVAL	CH0FVAL								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**FTMx\_FILTER field descriptions**

Field	Description
0–15 Reserved	This field is reserved.
16–19 CH3FVAL	Channel 3 Input Filter  Selects the filter value for the channel input.  The filter is disabled when the value is zero.

*Table continues on the next page...*

**FTMx\_FILTER field descriptions (continued)**

Field	Description
20–23 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
24–27 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
28–31 CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

**23.4.20 Fault Control (FTMx\_FLTCTRL)**

This register selects the filter value for the fault inputs, enables the fault inputs and the fault inputs filter.

Address: Base address + 7Ch offset

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R									0								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R				0						FFLTR3EN	FFLTR2EN	FFLTR1EN	FFLTR0EN	FAULT3EN	FAULT2EN	FAULT1EN	FAULT0EN
W										0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**FTMx\_FLTCTRL field descriptions**

Field	Description
0–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20–23 FFVAL	Fault Input Filter Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero.

*Table continues on the next page...*

**FTMx\_FLTCTRL field descriptions (continued)**

Field	Description
	<b>NOTE:</b> Writing to this field has immediate effect and must be done only when the fault control or all fault inputs are disabled. Failure to do this could result in a missing fault detection.
24 FFLTR3EN	Fault Input 3 Filter Enable  Enables the filter for the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input filter is disabled. 1 Fault input filter is enabled.
25 FFLTR2EN	Fault Input 2 Filter Enable  Enables the filter for the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input filter is disabled. 1 Fault input filter is enabled.
26 FFLTR1EN	Fault Input 1 Filter Enable  Enables the filter for the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input filter is disabled. 1 Fault input filter is enabled.
27 FFLTR0EN	Fault Input 0 Filter Enable  Enables the filter for the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input filter is disabled. 1 Fault input filter is enabled.
28 FAULT3EN	Fault Input 3 Enable  Enables the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input is disabled. 1 Fault input is enabled.
29 FAULT2EN	Fault Input 2 Enable  Enables the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input is disabled. 1 Fault input is enabled.
30 FAULT1EN	Fault Input 1 Enable  Enables the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.

*Table continues on the next page...*

**FTMx\_FLTCTRL field descriptions (continued)**

Field	Description
	0 Fault input is disabled. 1 Fault input is enabled.
31 FAULT0EN	Fault Input 0 Enable  Enables the fault input.  This field is write protected. It can be written only when MODE[WPDIS] = 1.  0 Fault input is disabled. 1 Fault input is enabled.

**23.4.21 Quadrature Decoder Control And Status (FTMx\_QDCTRL)**

This register has the control and status bits for the Quadrature Decoder mode.

Address: Base address + 80h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									PHAFLTREN	PHBFLTREN	PHAPOL	PHBPOL	QUADMODE	QUADIR	TOFDIR	QUADEN
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_QDCTRL field descriptions**

Field	Description
0–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 PHAFLTREN	Phase A Input Filter Enable  Enables the filter for the quadrature decoder phase A input. The filter value for the phase A input is defined by the CH0FVAL field of FILTER. The phase A filter is also disabled when CH0FVAL is zero.  0 Phase A input filter is disabled. 1 Phase A input filter is enabled.
25 PHBFLTREN	Phase B Input Filter Enable  Enables the filter for the quadrature decoder phase B input. The filter value for the phase B input is defined by the CH1FVAL field of FILTER. The phase B filter is also disabled when CH1FVAL is zero.  0 Phase B input filter is disabled. 1 Phase B input filter is enabled.
26 PHAPOL	Phase A Input Polarity  Selects the polarity for the quadrature decoder phase A input.  0 Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal.
27 PHBPOL	Phase B Input Polarity  Selects the polarity for the quadrature decoder phase B input.  0 Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal.
28 QUADMODE	Quadrature Decoder Mode  Selects the encoding mode used in the Quadrature Decoder mode.  0 Phase A and phase B encoding mode. 1 Count and direction encoding mode.
29 QUADIR	FTM Counter Direction In Quadrature Decoder Mode  Indicates the counting direction.  0 Counting direction is decreasing (FTM counter decrement). 1 Counting direction is increasing (FTM counter increment).
30 TOFDIR	Timer Overflow Direction In Quadrature Decoder Mode  Indicates if the TOF bit was set on the top or the bottom of counting.  0 TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (CNTIN register) to its maximum value (MOD register). 1 TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (CNTIN register).

*Table continues on the next page...*

**FTMx\_QDCTRL field descriptions (continued)**

Field	Description
31 QUADEN	<p>Quadrature Decoder Mode Enable</p> <p>Enables the Quadrature Decoder mode. In this mode, the phase A and B input signals control the FTM counter direction. The Quadrature Decoder mode has precedence over the other modes. See <a href="#">Table 23-6</a>.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 Quadrature Decoder mode is disabled. 1 Quadrature Decoder mode is enabled.</p>

**23.4.22 Configuration (FTMx\_CONF)**

This register selects the number of times that the FTM counter overflow should occur before the TOF bit to be set, the FTM behavior in modes, the use of an external global time base, and the global time base signal generation.

Address: Base address + 84h offset

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R									0								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R				0		GTBEOUT	GTBEEN	0	0	0	0	0					
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**FTMx\_CONF field descriptions**

Field	Description
0–20 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
21 GTBEOUT	<p>Global Time Base Output</p> <p>Enables the global time base signal generation to other FTMs.</p> <p>0 A global time base signal generation is disabled. 1 A global time base signal generation is enabled.</p>
22 GTBEEN	Global Time Base Enable

*Table continues on the next page...*

**FTMx\_CONF field descriptions (continued)**

Field	Description
	Configures the FTM to use an external global time base signal that is generated by another FTM. 0 Use of an external global time base is disabled. 1 Use of an external global time base is enabled.
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–31 NUMTOF	TOF Frequency Selects the ratio between the number of counter overflows to the number of times the TOF bit is set. NUMTOF = 0: The TOF bit is set for each counter overflow. NUMTOF = 1: The TOF bit is set for the first counter overflow but not for the next overflow. NUMTOF = 2: The TOF bit is set for the first counter overflow but not for the next 2 overflows. NUMTOF = 3: The TOF bit is set for the first counter overflow but not for the next 3 overflows. This pattern continues up to a maximum of 31.

**23.4.23 FTM Fault Input Polarity (FTMx\_FLTPOL)**

This register defines the fault inputs polarity.

Address: Base address + 88h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									0				FLT3POL	FLT2POL	FLT1POL	FLT0POL
W													0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_FLTPOL field descriptions**

Field	Description
0–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

**FTMx\_FLTPOL field descriptions (continued)**

Field	Description
28 FLT3POL	<p>Fault Input 3 Polarity</p> <p>Defines the polarity of the fault input.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.</p>
29 FLT2POL	<p>Fault Input 2 Polarity</p> <p>Defines the polarity of the fault input.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.</p>
30 FLT1POL	<p>Fault Input 1 Polarity</p> <p>Defines the polarity of the fault input.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.</p>
31 FLT0POL	<p>Fault Input 0 Polarity</p> <p>Defines the polarity of the fault input.</p> <p>This field is write protected. It can be written only when MODE[WPDIS] = 1.</p> <p>0 The fault input polarity is active high. A 1 at the fault input indicates a fault. 1 The fault input polarity is active low. A 0 at the fault input indicates a fault.</p>

### 23.4.24 Synchronization Configuration (FTMx\_SYNCONF)

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit, where  $j = 0, 1, 2$ , when the hardware trigger j is detected.

Address: Base address + 8Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W												HWSOC	HWINVC	HWOM	HWRBUF	HWRSTCNT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				0	SWSOC	SWINVC	SWOM	SWRBUF	SWRSTCNT	SYNCODE	0	SWOC	INVC	0	CNTINC	0
W											0	0	0	0	0	HWTRIGMOD E
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### FTMx\_SYNCONF field descriptions

Field	Description
0–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11 HWSOC	Software output control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the SWOCTRL register synchronization. 1 A hardware trigger activates the SWOCTRL register synchronization.
12 HWINVC	Inverting control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the INVCTRL register synchronization. 1 A hardware trigger activates the INVCTRL register synchronization.
13 HWOM	Output mask synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the OUTMASK register synchronization. 1 A hardware trigger activates the OUTMASK register synchronization.
14 HWRBUF	MOD, CNTIN, and CV registers synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 A hardware trigger activates MOD, CNTIN, and CV registers synchronization.
15 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the FTM counter synchronization. 1 A hardware trigger activates the FTM counter synchronization.

Table continues on the next page...

**FTMx\_SYNCONF field descriptions (continued)**

Field	Description
16–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19 SWSOC	Software output control synchronization is activated by the software trigger. 0 The software trigger does not activate the SWOCTRL register synchronization. 1 The software trigger activates the SWOCTRL register synchronization.
20 SWINVC	Inverting control synchronization is activated by the software trigger. 0 The software trigger does not activate the INVCTRL register synchronization. 1 The software trigger activates the INVCTRL register synchronization.
21 SWOM	Output mask synchronization is activated by the software trigger. 0 The software trigger does not activate the OUTMASK register synchronization. 1 The software trigger activates the OUTMASK register synchronization.
22 SWWRBUF	MOD, CNTIN, and CV registers synchronization is activated by the software trigger. 0 The software trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 The software trigger activates MOD, CNTIN, and CV registers synchronization.
23 SWRSTCNT	FTM counter synchronization is activated by the software trigger. 0 The software trigger does not activate the FTM counter synchronization. 1 The software trigger activates the FTM counter synchronization.
24 SYNCMODE	Synchronization Mode Selects the PWM Synchronization mode. 0 Legacy PWM synchronization is selected. 1 Enhanced PWM synchronization is selected.
25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 SWOC	SWOCTRL Register Synchronization 0 SWOCTRL register is updated with its buffer value at all rising edges of system clock. 1 SWOCTRL register is updated with its buffer value by the PWM synchronization.
27 INVNC	INVCTRL Register Synchronization 0 INVCTRL register is updated with its buffer value at all rising edges of system clock. 1 INVCTRL register is updated with its buffer value by the PWM synchronization.
28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29 CNTINC	CNTIN Register Synchronization 0 CNTIN register is updated with its buffer value at all rising edges of system clock. 1 CNTIN register is updated with its buffer value by the PWM synchronization.
30 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
31 HWTRIGMODE	Hardware Trigger Mode

*Table continues on the next page...*

**FTMx\_SYNCONF field descriptions (continued)**

Field	Description
0	FTM clears the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.
1	FTM does not clear the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.

**23.4.25 FTM Inverting Control (FTMx\_INVCTRL)**

This register controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

Address: Base address + 90h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									0							
W													INV3EN	INV2EN	INV1EN	INV0EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_INVCTRL field descriptions**

Field	Description
0–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 INV3EN	Pair Channels 3 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
29 INV2EN	Pair Channels 2 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
30 INV1EN	Pair Channels 1 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

*Table continues on the next page...*

**FTMx\_INVCTRL field descriptions (continued)**

Field	Description
31 INV0EN	Pair Channels 0 Inverting Enable  0 Inverting is disabled. 1 Inverting is enabled.

**23.4.26 FTM Software Output Control (FTMx\_SWOCTRL)**

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CHnOC bits enable the control of the corresponding channel (n) output by software.
- The CHnOCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Address: Base address + 94h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CH7OCV	CH6OCV	CH5OCV	CH4OCV	CH3OCV	CH2OCV	CH1OCV	CH0OCV	CH7OC	CH6OC	CH5OC	CH4OC	CH3OC	CH2OC	CH1OC	CH0OC
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FTMx\_SWOCTRL field descriptions**

Field	Description
0–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 CH7OCV	Channel 7 Software Output Control Value  0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
17 CH6OCV	Channel 6 Software Output Control Value

*Table continues on the next page...*

**FTMx\_SWOCTRL field descriptions (continued)**

Field	Description
	0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
18 CH5OCV	Channel 5 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
19 CH4OCV	Channel 4 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
20 CH3OCV	Channel 3 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
21 CH2OCV	Channel 2 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
22 CH1OCV	Channel 1 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
23 CH0OCV	Channel 0 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
24 CH7OC	Channel 7 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
25 CH6OC	Channel 6 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
26 CH5OC	Channel 5 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
27 CH4OC	Channel 4 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
28 CH3OC	Channel 3 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
29 CH2OC	Channel 2 Software Output Control Enable

*Table continues on the next page...*

**FTMx\_SWOCTRL field descriptions (continued)**

Field	Description
	0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
30 CH1OC	Channel 1 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
31 CH0OC	Channel 0 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.

**23.4.27 FTM PWM Load (FTMx\_PWMLOAD)**

Enables the loading of the MOD, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for the channel (j) when FTM counter = C(j)V.

Address: Base address + 98h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R									0								
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R				0			LDOK		0	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL
W									0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**FTMx\_PWMLOAD field descriptions**

Field	Description
0–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 LDOK	Load Enable Enables the loading of the MOD, CNTIN, and CV registers with the values of their write buffers. 0 Loading updated values is disabled. 1 Loading updated values is enabled.

Table continues on the next page...

**FTMx\_PWMLOAD field descriptions (continued)**

Field	Description
23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24 CH7SEL	Channel 7 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
25 CH6SEL	Channel 6 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
26 CH5SEL	Channel 5 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
27 CH4SEL	Channel 4 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
28 CH3SEL	Channel 3 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
29 CH2SEL	Channel 2 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
30 CH1SEL	Channel 1 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
31 CH0SEL	Channel 0 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.

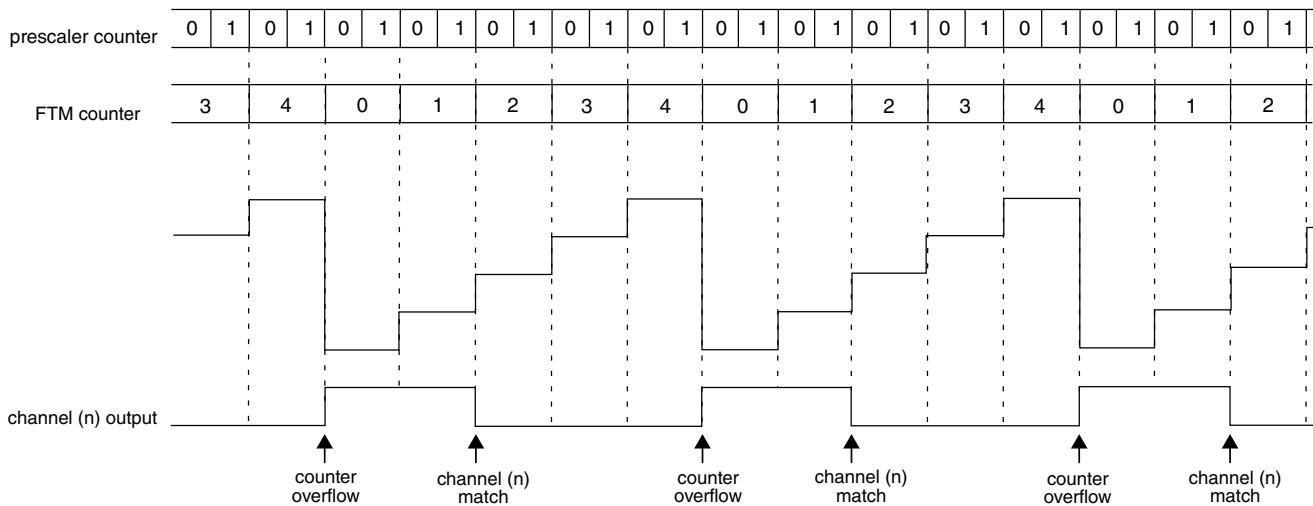
## 23.5 Functional description

The notation used in this document to represent the counters and the generation of the signals is shown in the following figure.

## Functional description

FTM counting is up.  
Channel (n) is in high-true EPWM mode.

PS[2:0] = 001  
CNTIN = 0x0000  
MOD = 0x0004  
CnV = 0x0002



**Figure 23-2. Notation used**

### 23.5.1 Clock source

The FTM has only one clock domain: the system clock.

#### 23.5.1.1 Counter clock source

The CLKS[1:0] bits in the SC register selects clock sources for the FTM counter or disables the FTM counter. After any chip reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the system clock or an external clock. This clock input is defined by chip integration; see the chip-specific FTM information for further details. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the system clock frequency.

The external clock passes through a synchronizer clocked by the system clock to assure that counter transitions are properly aligned to system clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

## 23.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and FTM counter.

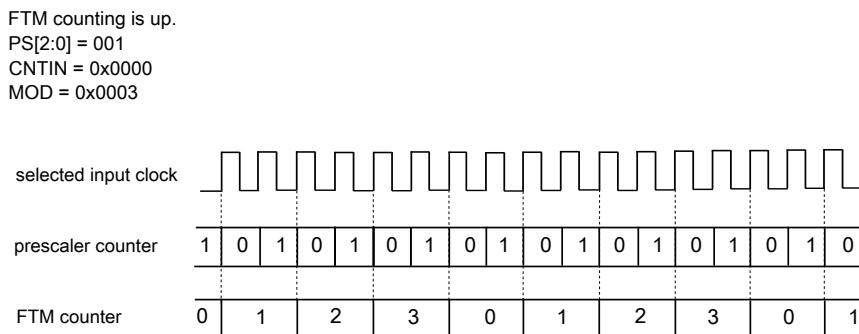


Figure 23-3. Example of the prescaler counter

## 23.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- Up counting
- Up-down counting
- Quadrature Decoder mode

### 23.5.3.1 Up counting

Up counting is selected when:

- QUADEN = 0, and
- CPWMS = 0

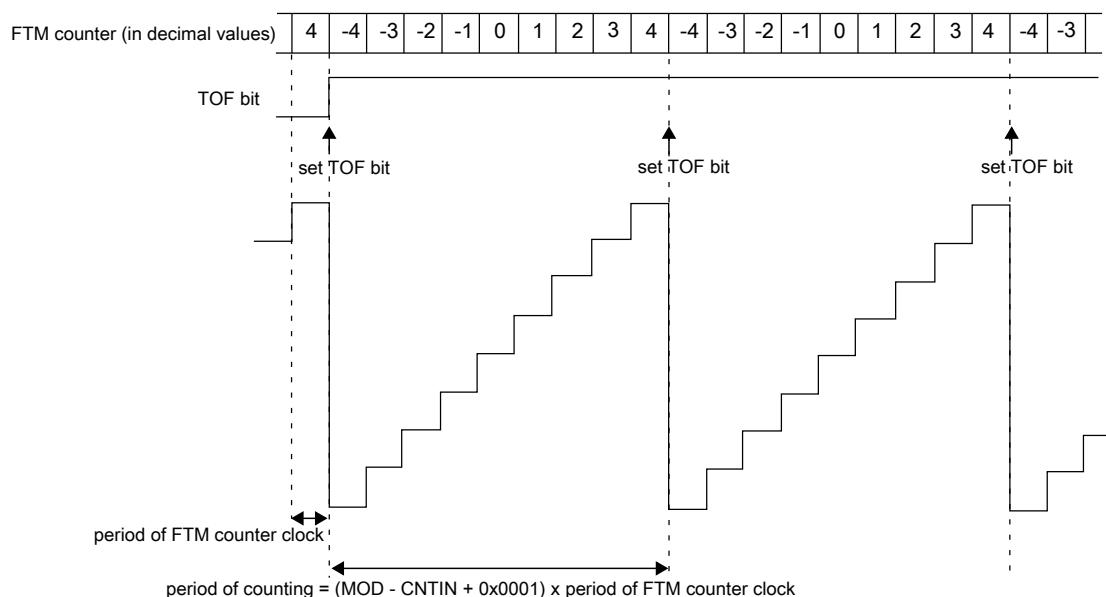
## Functional description

CNTIN defines the starting value of the count and MOD defines the final value of the count, see the following figure. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is  $(MOD - CNTIN + 0x0001) \times \text{period of the FTM counter clock}$ .

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

FTM counting is up.  
 CNTIN = 0xFFFF (in two's complement is equal to -4)  
 MOD = 0x0004



**Figure 23-4. Example of FTM up and signed counting**

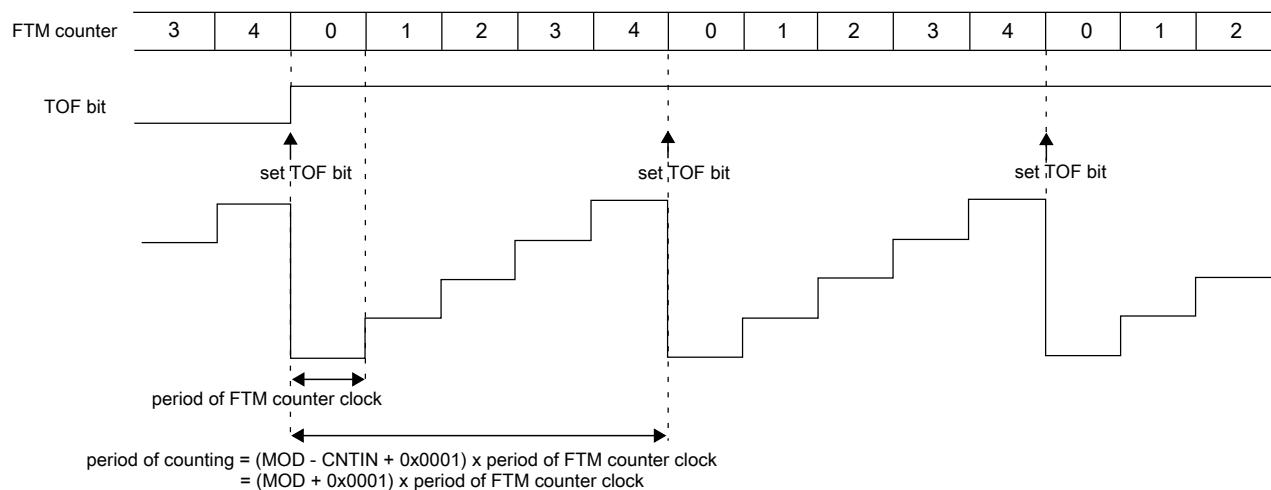
**Table 23-8. FTM counting based on CNTIN value**

When	Then
CNTIN = 0x0000	The FTM counting is equivalent to TPM up counting, that is, up and unsigned counting. See the following figure.
CNTIN[15] = 1	The initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed.
CNTIN[15] = 0 and CNTIN ≠ 0x0000	The initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

FTM counting is up

CNTIN = 0x0000

MOD = 0x0004

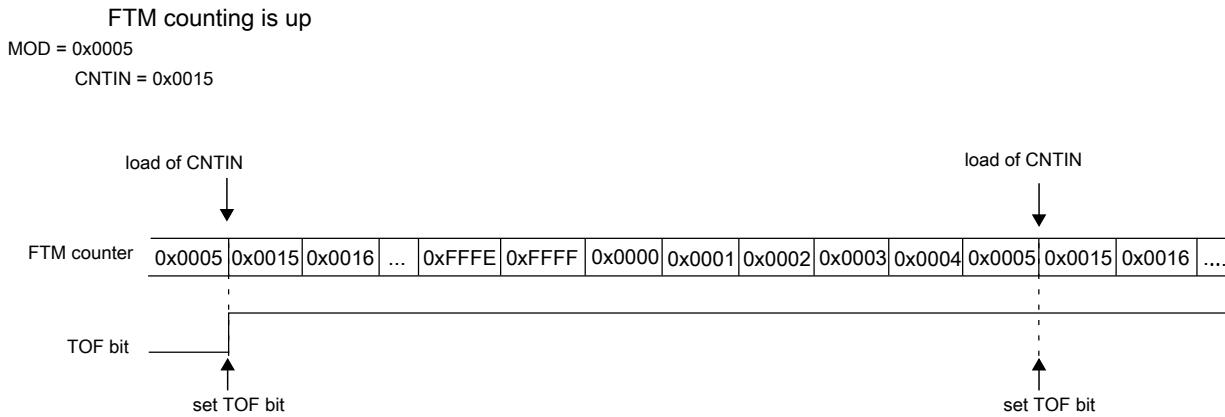


**Figure 23-5. Example of FTM up counting with CNTIN = 0x0000**

### Note

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register, either in the unsigned counting or signed counting. It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000, for example after reset, and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

## Functional description



**Figure 23-6. Example of up counting when the value of CNTIN is greater than the value of MOD**

### 23.5.3.2 Up-down counting

Up-down counting is selected when:

- QUADEN = 0, and
- CPWMS = 1

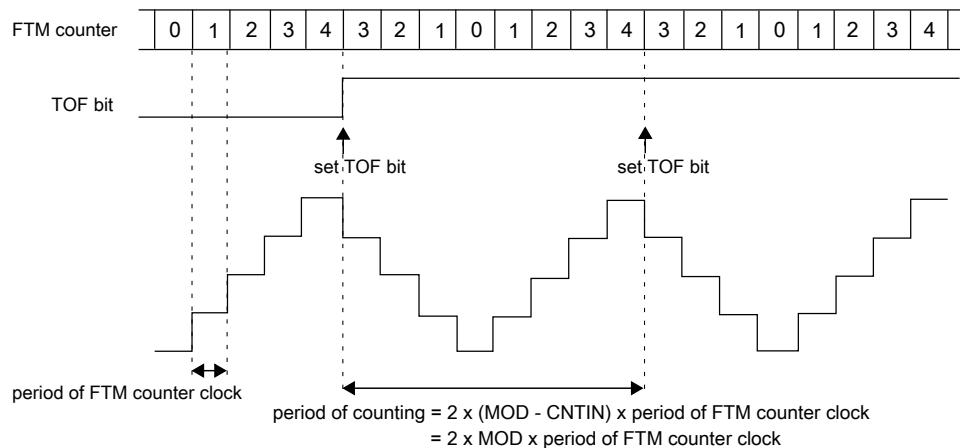
CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is  $2 \times (\text{MOD} - \text{CNTIN}) \times \text{period of the FTM counter clock}$ .

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting, that is, up-down and unsigned counting. See the following figure.

FTM counting is up-down  
CNTIN = 0x0000  
MOD = 0x0004



**Figure 23-7. Example of up-down counting when CNTIN = 0x0000**

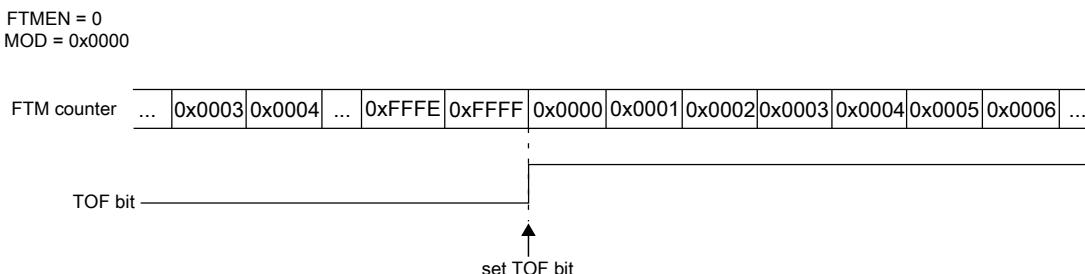
### Note

When CNTIN is different from zero in the up-down counting, a valid CPWM signal is generated:

- if  $CnV > CNTIN$ , or
- if  $CnV = 0$  or if  $CnV[15] = 1$ . In this case, 0% CPWM is generated.

### 23.5.3.3 Free running counter

If ( $FTMEN = 0$ ) and ( $\text{MOD} = 0x0000$  or  $\text{MOD} = 0xFFFF$ ), the FTM counter is a free running counter. In this case, the FTM counter runs free from  $0x0000$  through  $0xFFFF$  and the TOF bit is set when the FTM counter changes from  $0xFFFF$  to  $0x0000$ . See the following figure.



**Figure 23-8. Example when the FTM counter is free running**

The FTM counter is also a free running counter when:

- FTMEN = 1
- QUADEN = 0
- CPWMS = 0
- CNTIN = 0x0000, and
- MOD = 0xFFFF

### 23.5.3.4 Counter reset

Any one of the following cases resets the FTM counter to the value in the CNTIN register and the channels output to its initial value, except for channels in Output Compare mode.

- Any write to CNT.
- **FTM counter synchronization.**

### 23.5.3.5 When the TOF bit is set

The NUMTOF[4:0] bits define the number of times that the FTM counter overflow should occur before the TOF bit to be set. If NUMTOF[4:0] = 0x00, then the TOF bit is set at each FTM counter overflow.

Initialize the FTM counter, by writing to CNT, after writing to the NUMTOF[4:0] bits to avoid confusion about when the first counter overflow will occur.

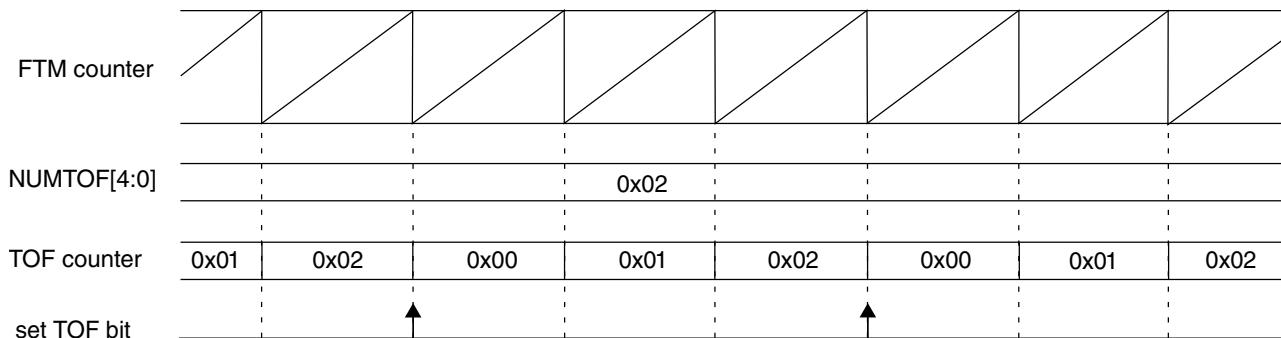
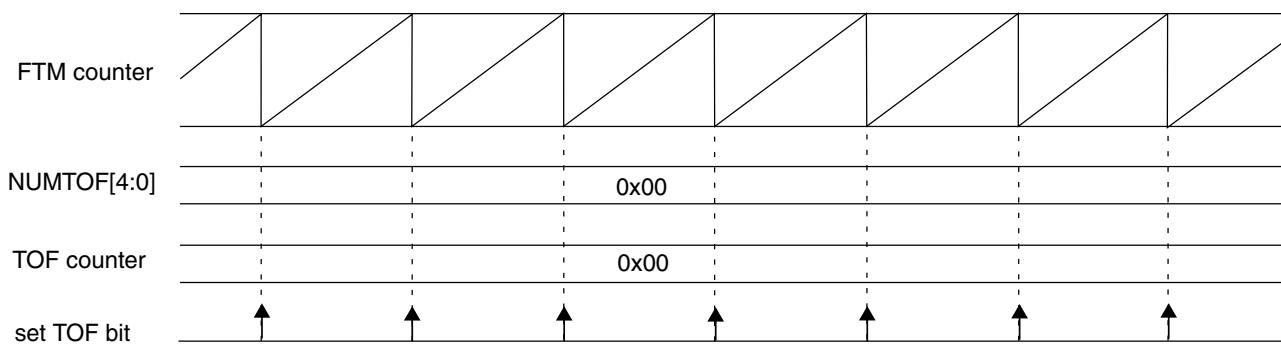


Figure 23-9. Periodic TOF when NUMTOF = 0x02



**Figure 23-10. Periodic TOF when NUMTOF = 0x00**

### 23.5.4 Input Capture mode

The Input Capture mode is selected when:

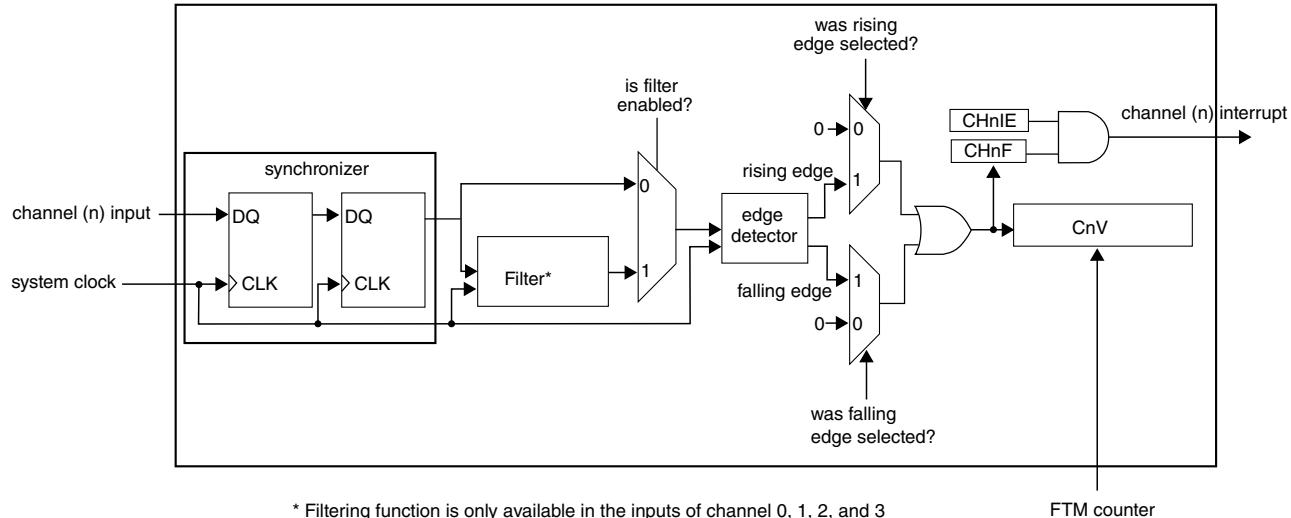
- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0
- MSnB:MSnA = 0:0, and
- ELSnB:ELSnA ≠ 0:0

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1. See the following figure.

When a channel is configured for input capture, the FTMxCHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is system clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register is ignored in Input Capture mode.

While in , the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of , is captured into the CnV register and the CHnF bit is set.

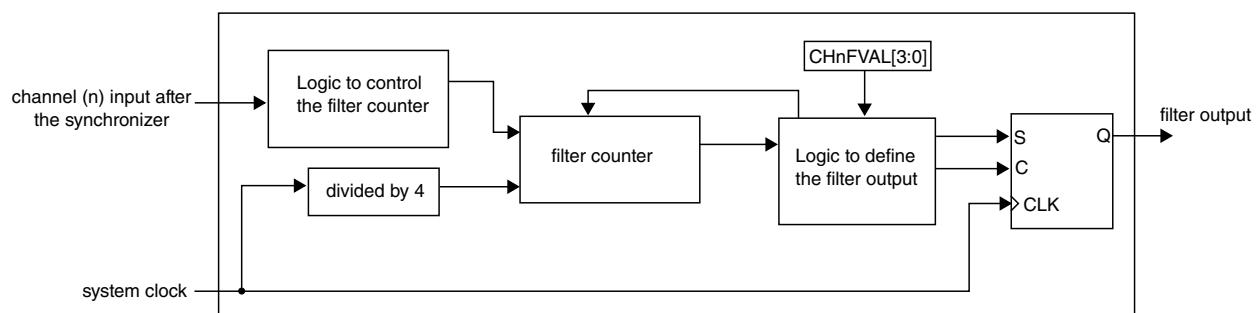
**Figure 23-11. Input Capture mode**

If the channel input does not have a filter enabled, then the input signal is always delayed 3 rising edges of the system clock, that is, two rising edges to the synchronizer plus one more rising edge to the edge detector. In other words, the CHnF bit is set on the third rising edge of the system clock after a valid edge occurs on the channel input.

### 23.5.4.1 Filter for Input Capture mode

The filter function is only available on channels 0, 1, 2, and 3.

First, the input signal is synchronized by the system clock. Following synchronization, the input signal enters the filter block. See the following figure.

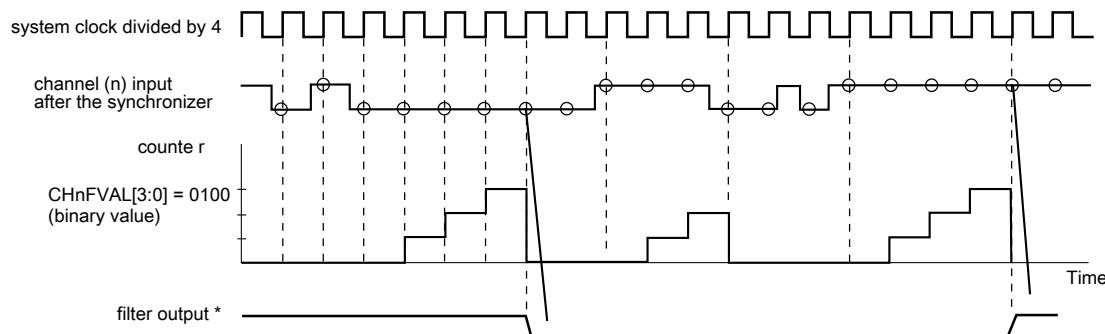
**Figure 23-12. Channel input filter**

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to CHnFVAL[3:0], the state change of the input signal is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by CHnFVAL[3:0] ( $\times 4$  system clocks) is regarded as a glitch and is not passed on to the edge detector. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed 3 rising edges of the system clock. If (CHnFVAL[3:0]  $\neq$  0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0]  $\times$  4 system clocks) plus a further 4 rising edges of the system clock: two rising edges to the synchronizer, one rising edge to the filter output, plus one more to the edge detector. In other words, CHnF is set ( $4 + 4 \times \text{CHnFVAL}[3:0]$ ) system clock periods after a valid edge occurs on the channel input.

The clock for the counter in the channel input filter is the system clock divided by 4.



\* Note: Filter output is delayed one system clock of filter counter logic output.

**Figure 23-13. Channel input filter example**

The figure below shows an example of input capture with filter enabled and the delay added by each part of the input capture logic. Note that the input signal is delayed only by the synchronizer and edge detector logic if the filter is disabled.

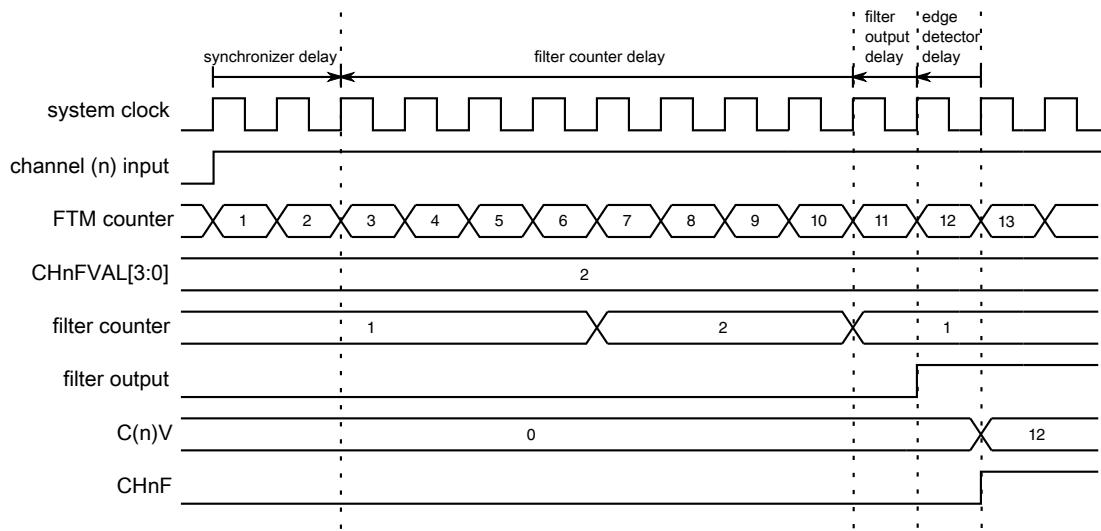


Figure 23-14. Input capture example

### 23.5.5 Output Compare mode

The Output Compare mode is selected when:

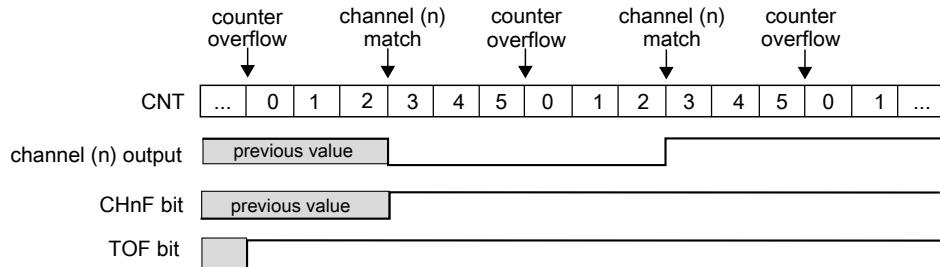
- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0, and
- MSnB:MSnA = 0:1

In Output Compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs.

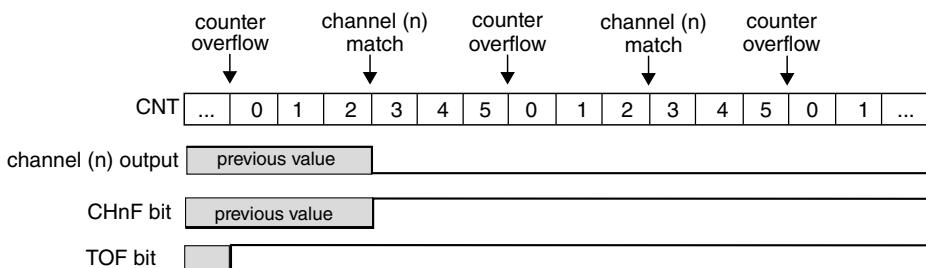
The CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1 at the channel (n) match (FTM counter = CnV).

MOD = 0x0005  
CnV = 0x0003



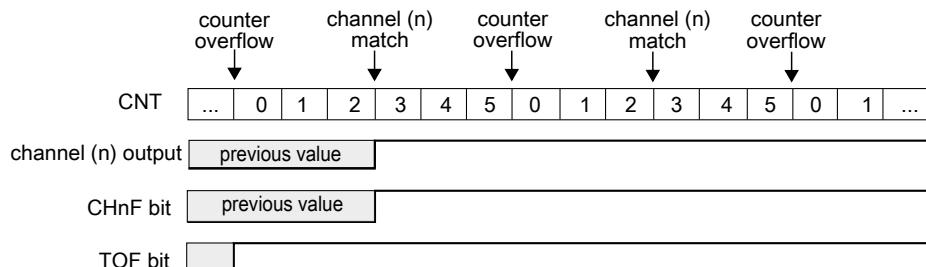
**Figure 23-15. Example of the Output Compare mode when the match toggles the channel output**

MOD = 0x0005  
CnV = 0x0003



**Figure 23-16. Example of the Output Compare mode when the match clears the channel output**

MOD = 0x0005  
CnV = 0x0003



**Figure 23-17. Example of the Output Compare mode when the match sets the channel output**

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1, however the channel (n) output is not modified and controlled by FTM.

### 23.5.6 Edge-Aligned PWM (EPWM) mode

The Edge-Aligned mode is selected when:

- QUADEN = 0

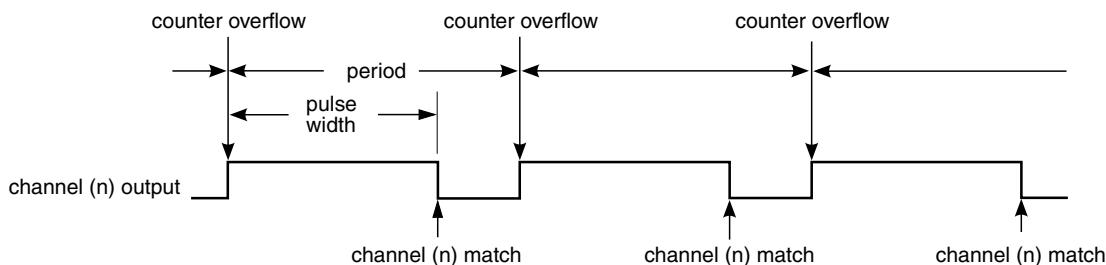
## Functional description

- DECAPEN = 0
- COMBINE = 0
- CPWMS = 0, and
- MSnB = 1

The EPWM period is determined by (MOD – CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV – CNTIN).

The CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1 at the channel (n) match (FTM counter = CnV), that is, at the end of the pulse width.

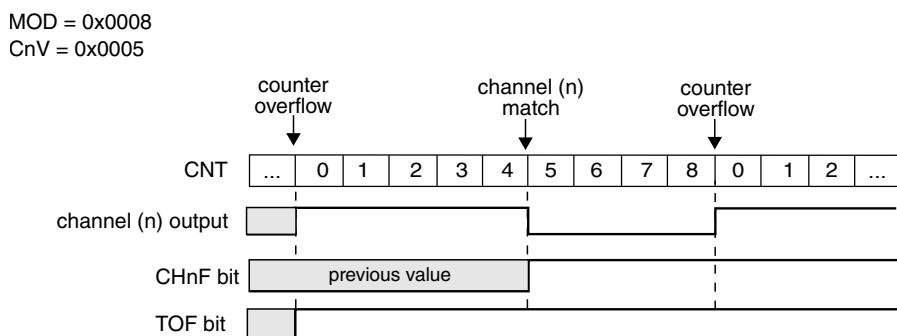
This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.



**Figure 23-18. EPWM period and pulse width with ELSnB:ELSnA = 1:0**

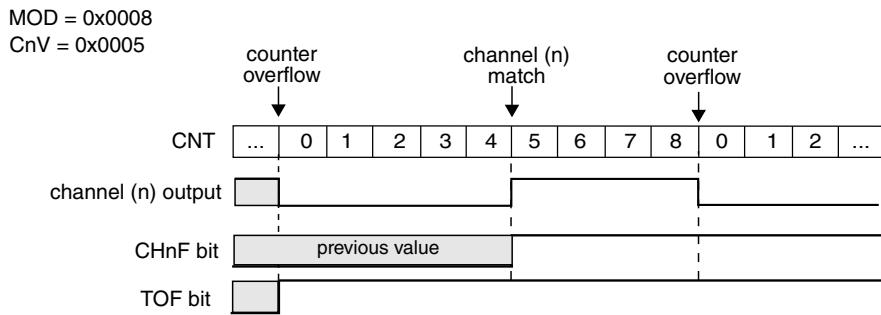
If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated if CHnIE = 1, however the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced low at the channel (n) match (FTM counter = CnV). See the following figure.



**Figure 23-19. EPWM signal with ELSnB:ELSnA = 1:0**

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced high at the channel (n) match (FTM counter = CnV). See the following figure.



**Figure 23-20. EPWM signal with ELSnB:ELSnA = X:1**

If ( $CnV = 0x0000$ ), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set even when there is the channel (n) match.

If ( $CnV > MOD$ ), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

### Note

When CNTIN is different from zero the following EPWM signals can be generated:

- 0% EPWM signal if  $CnV = CNTIN$ ,
- EPWM signal between 0% and 100% if  $CNTIN < CnV \leq MOD$ ,
- 100% EPWM signal when  $CNTIN > CnV$  or  $CnV > MOD$ .

## 23.5.7 Center-Aligned PWM (CPWM) mode

The Center-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 0, and
- CPWMS = 1

The CPWM pulse width (duty cycle) is determined by  $2 \times (CnV - CNTIN)$  and the period is determined by  $2 \times (MOD - CNTIN)$ . See the following figure. MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

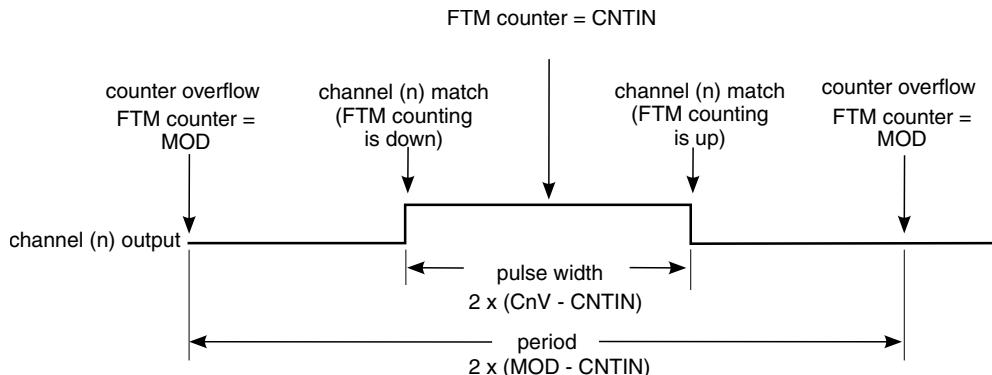
In the CPWM mode, the FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

## Functional description

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of CNTIN.

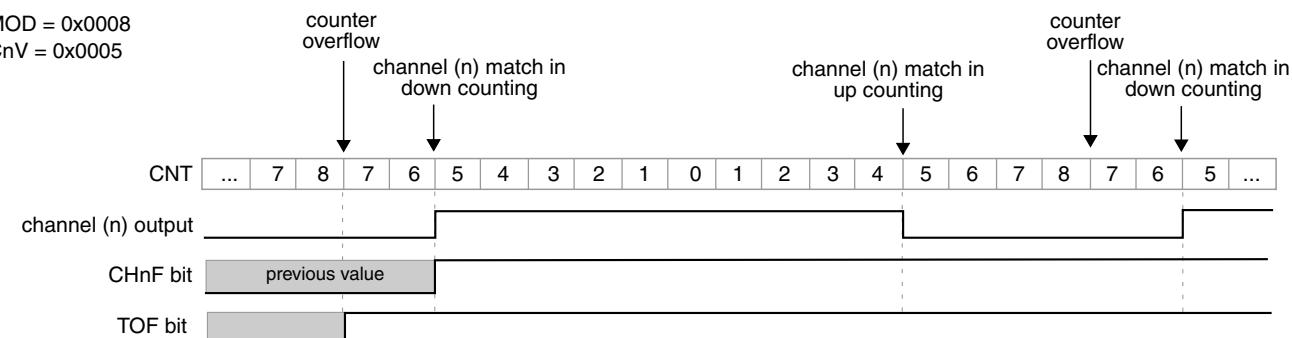
The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).



**Figure 23-21. CPWM period and pulse width with ELSnB:ELSnA = 1:0**

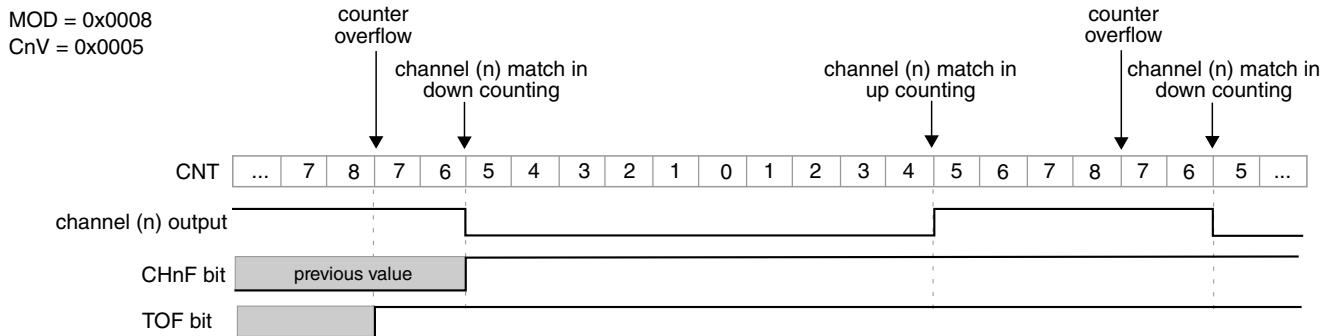
If (ELSnB:ELSnA = 0:0) when the FTM counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. See the following figure.



**Figure 23-22. CPWM signal with ELSnB:ELSnA = 1:0**

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. See the following figure.

**Figure 23-23. CPWM signal with ELSnB:ELSnA = X:1**

If ( $CnV = 0x0000$ ) or  $CnV$  is a negative value, that is ( $CnV[15] = 1$ ), then the channel (n) output is a 0% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match.

If  $CnV$  is a positive value, that is ( $CnV[15] = 0$ ), ( $CnV \geq MOD$ ), and ( $MOD \neq 0x0000$ ), then the channel (n) output is a 100% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE, 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

### 23.5.8 Combine mode

The Combine mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMBINE = 1, and
- CPWMS = 0

In Combine mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by ( $MOD - CNTIN + 0x0001$ ) and the PWM pulse width (duty cycle) is determined by ( $|C(n+1)V - C(n)V|$ ).

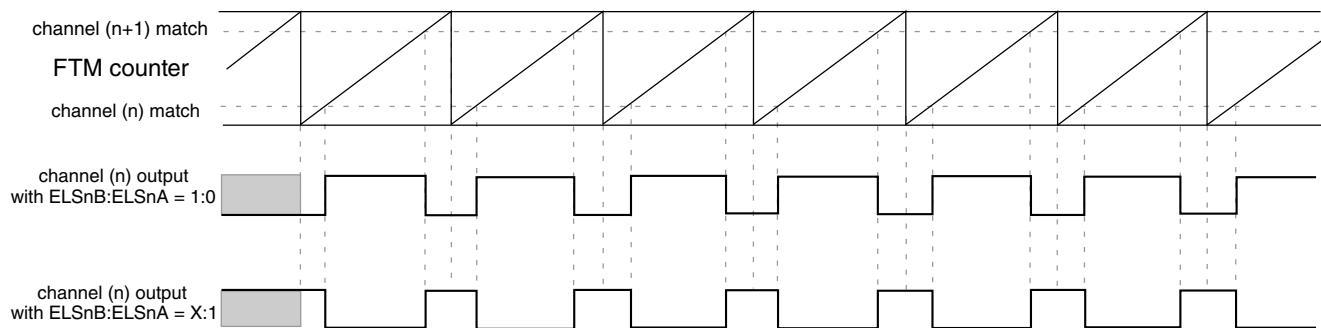
The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter =  $C(n)V$ ). The CH(n+1)F bit is set and the channel (n+1) interrupt is generated, if CH(n+1)IE = 1, at the channel (n+1) match (FTM counter =  $C(n+1)V$ ).

## Functional description

If ( $\text{ELSnB:ELSnA} = 1:0$ ), then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter =  $C(n+1)V$ ). It is forced high at the channel (n) match (FTM counter =  $C(n)V$ ). See the following figure.

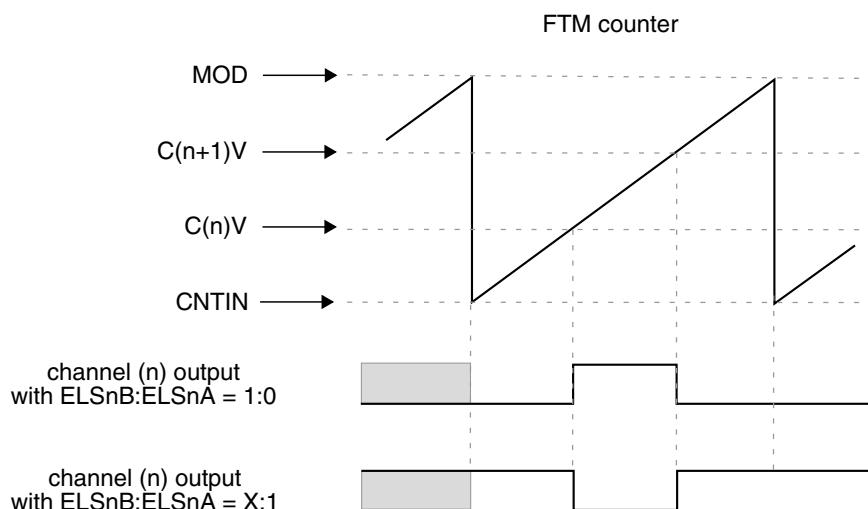
If ( $\text{ELSnB:ELSnA} = X:1$ ), then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter =  $C(n+1)V$ ). It is forced low at the channel (n) match (FTM counter =  $C(n)V$ ). See the following figure.

In Combine mode, the ELS(n+1)B and ELS(n+1)A bits are not used in the generation of the channels (n) and (n+1) output. However, if ( $\text{ELSnB:ELSnA} = 0:0$ ) then the channel (n) output is not controlled by FTM, and if ( $\text{ELS}(n+1)\text{B:ELS}(n+1)\text{A} = 0:0$ ) then the channel (n+1) output is not controlled by FTM.

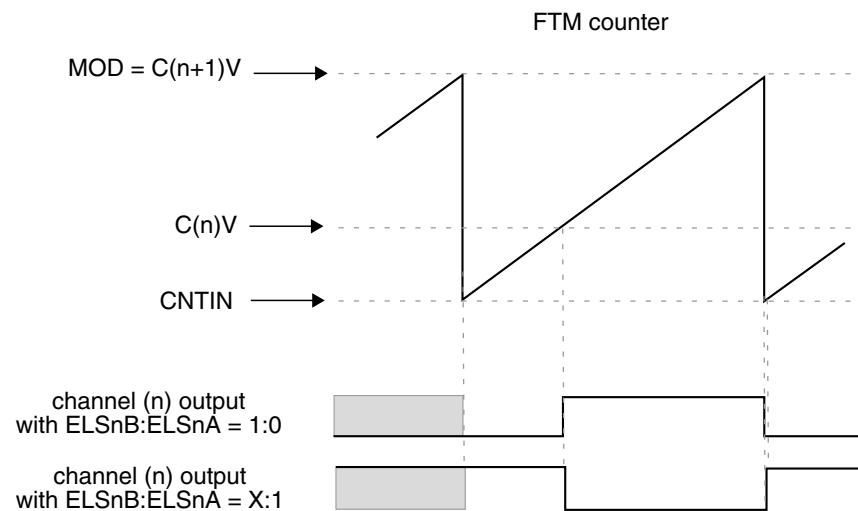


**Figure 23-24. Combine mode**

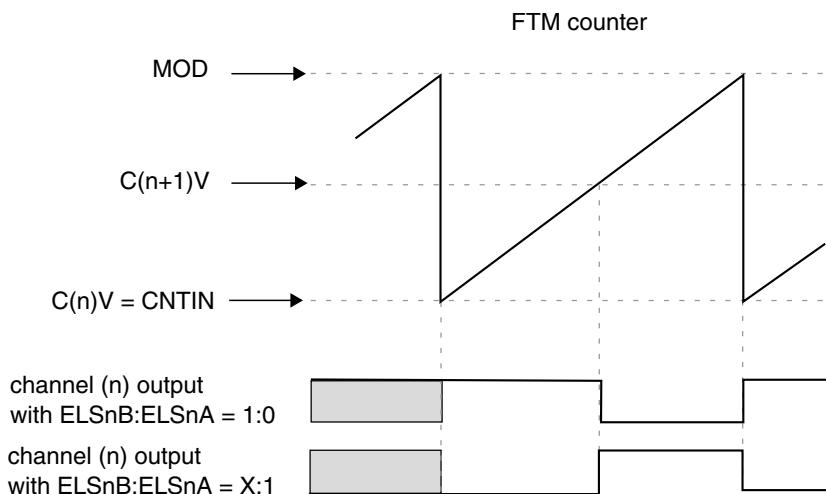
The following figures illustrate the PWM signals generation using Combine mode.



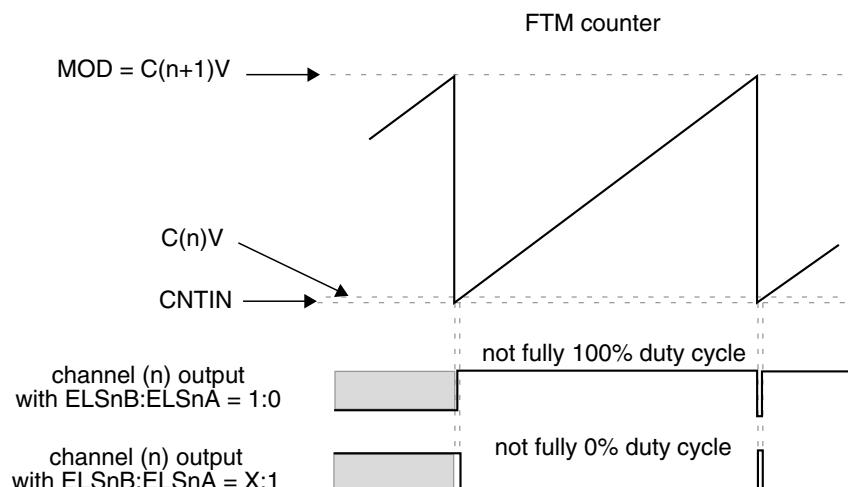
**Figure 23-25. Channel (n) output if ( $\text{CNTIN} < \text{C}(n)\text{V} < \text{MOD}$ ) and ( $\text{CNTIN} < \text{C}(n+1)\text{V} < \text{MOD}$ ) and ( $\text{C}(n)\text{V} < \text{C}(n+1)\text{V}$ )**



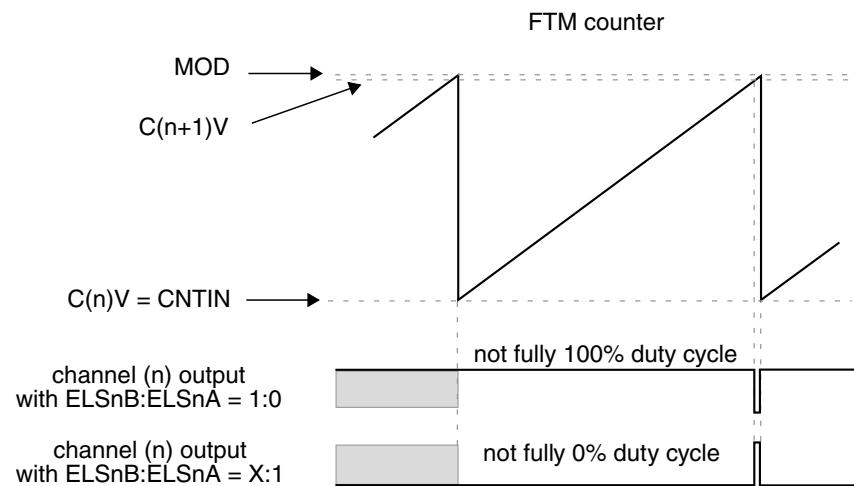
**Figure 23-26. Channel (n) output if ( $C(n)V < C(n+1)V < MOD$ ) and ( $C(n+1)V = MOD$ )**



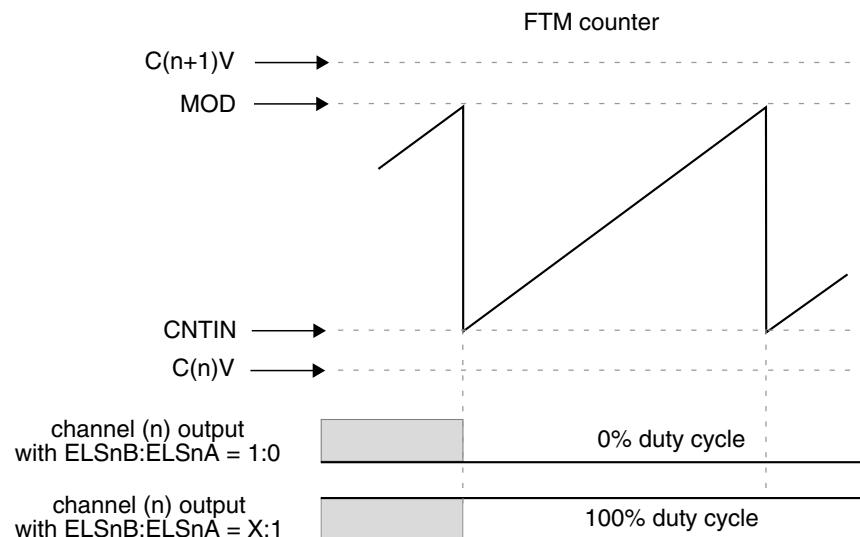
**Figure 23-27. Channel (n) output if ( $C(n)V = CNTIN$ ) and ( $CNTIN < C(n+1)V < MOD$ )**



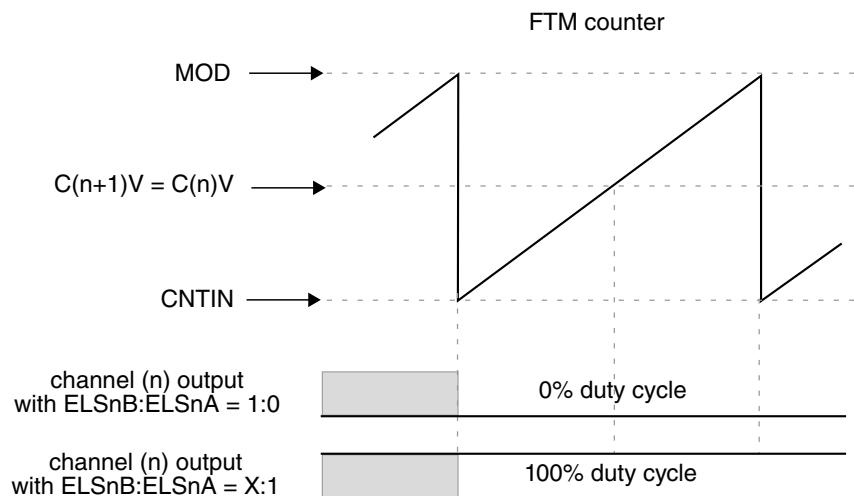
**Figure 23-28. Channel (n) output if ( $CNTIN < C(n)V < MOD$ ) and ( $C(n)V$  is Almost Equal to  $CNTIN$ ) and ( $C(n+1)V = MOD$ )**



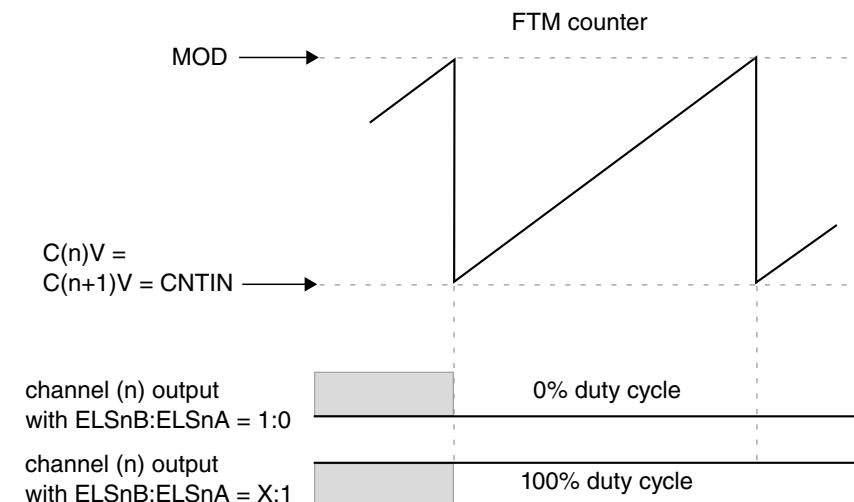
**Figure 23-29. Channel (n) output if ( $C(n)V = CNTIN$ ) and ( $CNTIN < C(n+1)V < MOD$ ) and ( $C(n+1)V$  is Almost Equal to MOD)**



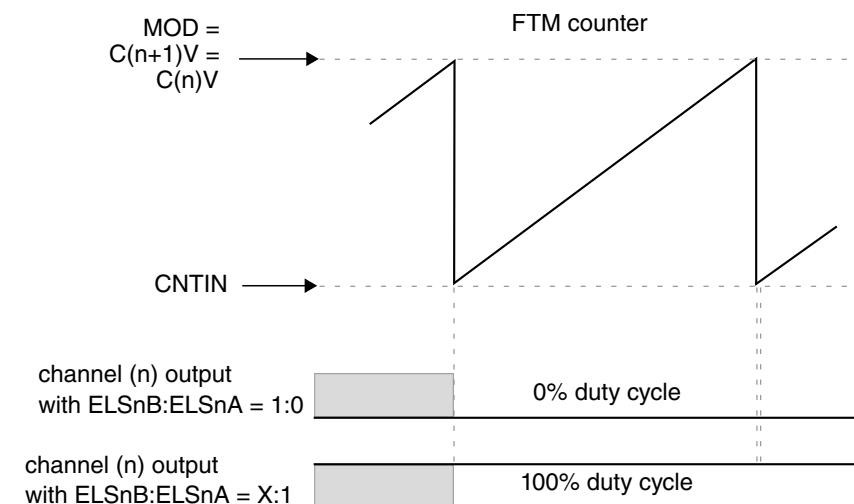
**Figure 23-30. Channel (n) output if  $C(n)V$  and  $C(n+1)V$  are not between  $CNTIN$  and  $MOD$**



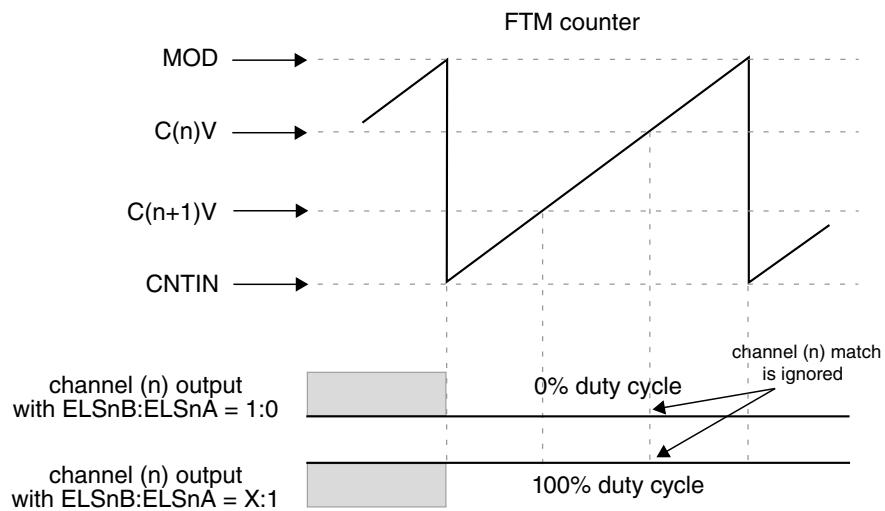
**Figure 23-31. Channel (n) output if ( $CNTIN < C(n)V < MOD$ ) and ( $CNTIN < C(n+1)V < MOD$ ) and ( $C(n)V = C(n+1)V$ )**



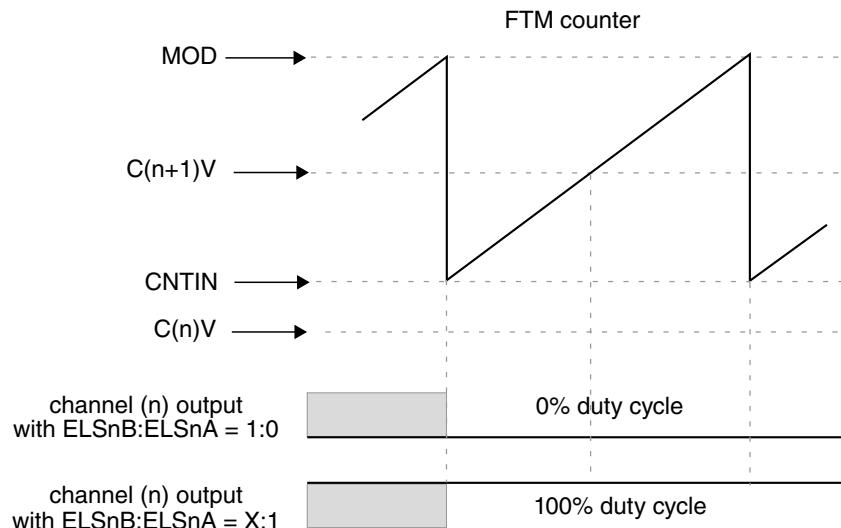
**Figure 23-32. Channel (n) output if ( $C(n)V = C(n+1)V = CNTIN$ )**



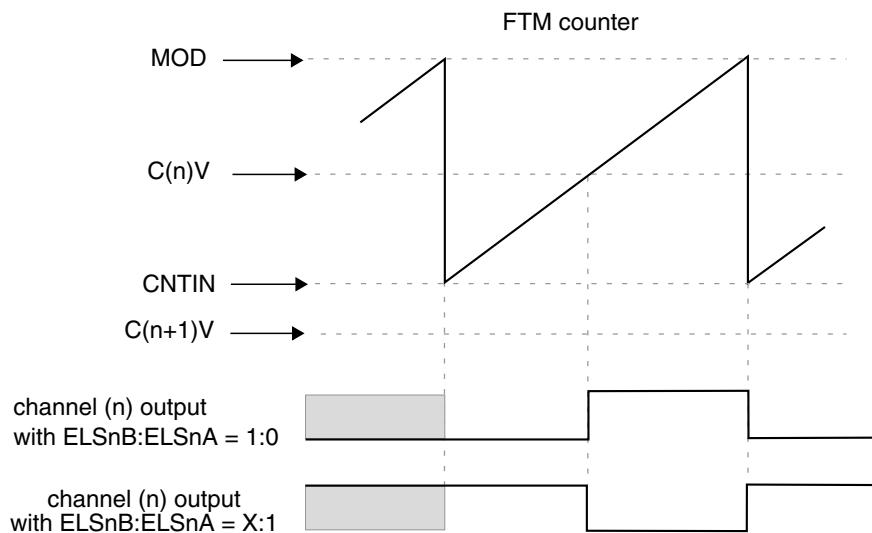
**Figure 23-33. Channel (n) output if ( $C(n)V = C(n+1)V = MOD$ )**



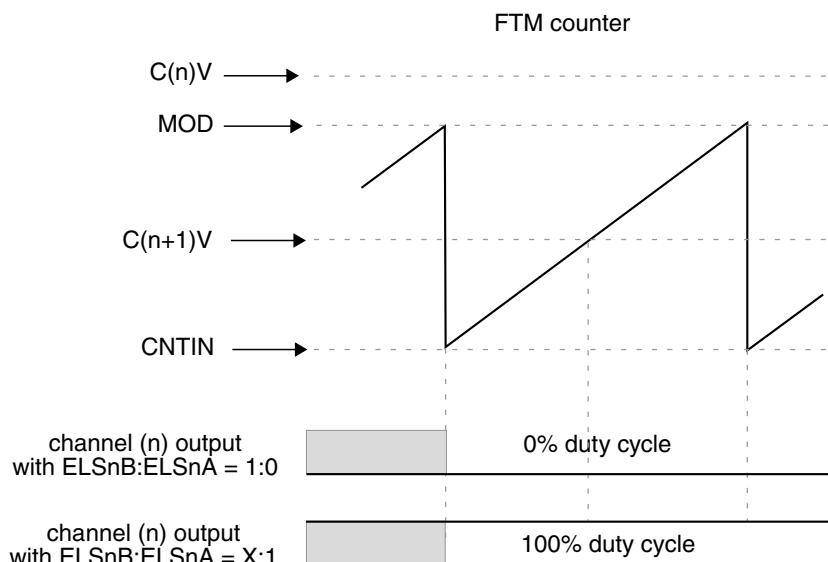
**Figure 23-34. Channel (n) output if ( $CNTIN < C(n)V < MOD$ ) and ( $CNTIN < C(n+1)V < MOD$ ) and ( $C(n)V > C(n+1)V$ )**



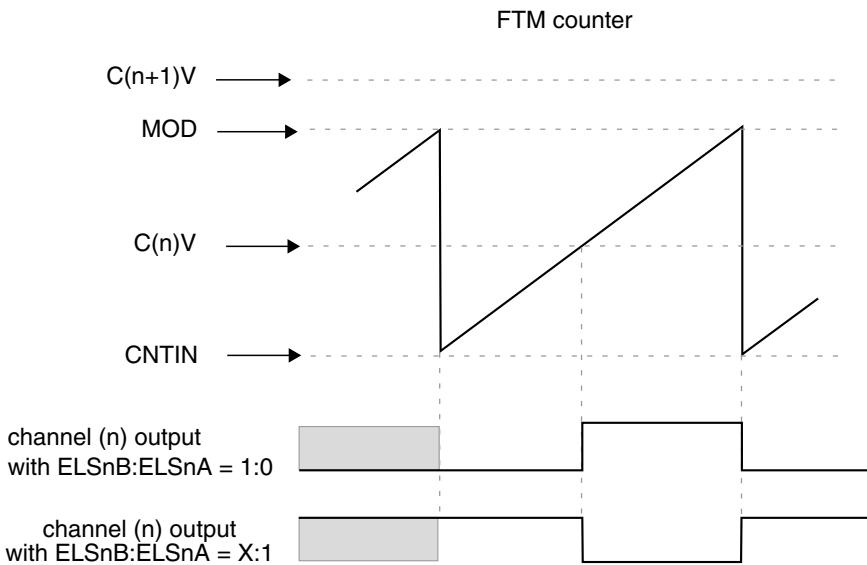
**Figure 23-35. Channel (n) output if ( $C(n)V < CNTIN$ ) and ( $CNTIN < C(n+1)V < MOD$ )**



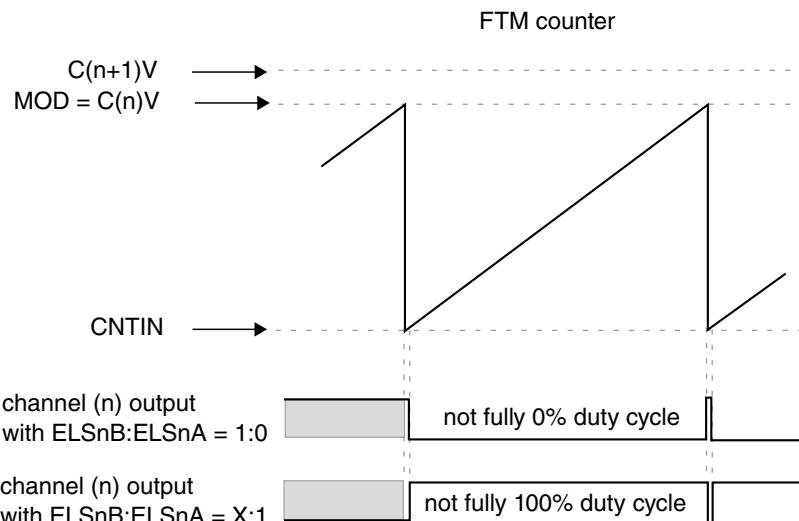
**Figure 23-36. Channel (n) output if  $(C(n+1)V < CNTIN)$  and  $(CNTIN < C(n)V < MOD)$**



**Figure 23-37. Channel (n) output if  $(C(n)V > MOD)$  and  $(CNTIN < C(n+1)V < MOD)$**



**Figure 23-38. Channel (n) output if ( $C(n+1)V > MOD$ ) and ( $CNTIN < C(n)V < MOD$ )**



**Figure 23-39. Channel (n) output if ( $C(n+1)V > MOD$ ) and ( $CNTIN < C(n)V = MOD$ )**

### 23.5.8.1 Asymmetrical PWM

In Combine mode, the control of the PWM signal first edge, when the channel (n) match occurs, that is,  $FTM$  counter  $= C(n)V$ , is independent of the control of the PWM signal second edge, when the channel (n+1) match occurs, that is,  $FTM$  counter  $= C(n+1)V$ . So, Combine mode allows the generation of asymmetrical PWM signals.

### 23.5.9 Complementary mode

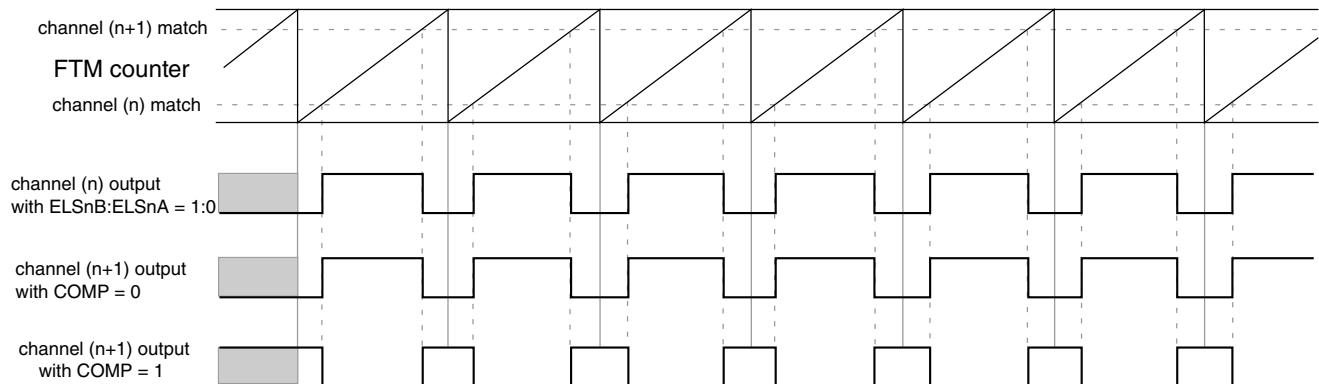
The Complementary mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1

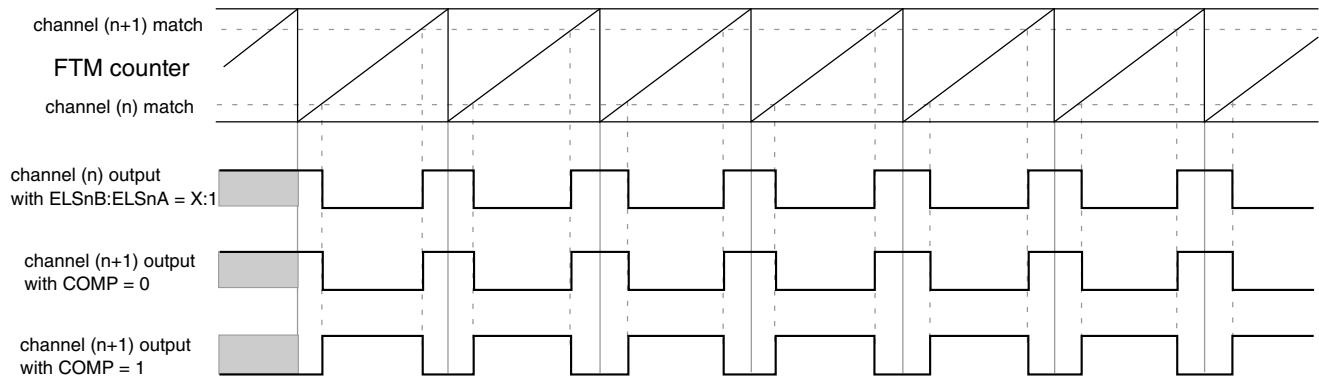
In Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

So, the channel (n+1) output is the same as the channel (n) output when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 0



**Figure 23-40. Channel (n+1) output in Complementary mode with (ELSnB:ELSnA = 1:0)**



**Figure 23-41. Channel (n+1) output in Complementary mode with (ELSnB:ELSnA = X:1)**

### NOTE

The complementary mode is not available in Output Compare mode.

#### 23.5.10 Registers updated from write buffers

### 23.5.10.1 CNTIN register update

The following table describes when CNTIN register is updated:

**Table 23-9. CNTIN register update**

When	Then CNTIN register is updated
CLKS[1:0] = 0:0	When CNTIN register is written, independent of FTMEN bit.
<ul style="list-style-type: none"> <li>• FTMEN = 0, or</li> <li>• CNTINC = 0</li> </ul>	At the next system clock after CNTIN was written.
<ul style="list-style-type: none"> <li>• FTMEN = 1,</li> <li>• SYNCMODE = 1, and</li> <li>• CNTINC = 1</li> </ul>	By the <a href="#">CNTIN register synchronization</a> .

### 23.5.10.2 MOD register update

The following table describes when MOD register is updated:

**Table 23-10. MOD register update**

When	Then MOD register is updated
CLKS[1:0] = 0:0	When MOD register is written, independent of FTMEN bit.
<ul style="list-style-type: none"> <li>• CLKS[1:0] ≠ 0:0, and</li> <li>• FTMEN = 0</li> </ul>	According to the CPWMS bit, that is: <ul style="list-style-type: none"> <li>• If the selected mode is not CPWM then MOD register is updated after MOD register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000.</li> <li>• If the selected mode is CPWM then MOD register is updated after MOD register was written and the FTM counter changes from MOD to (MOD – 0x0001).</li> </ul>
<ul style="list-style-type: none"> <li>• CLKS[1:0] ≠ 0:0, and</li> <li>• FTMEN = 1</li> </ul>	By the <a href="#">MOD register synchronization</a> .

### 23.5.10.3 CnV register update

The following table describes when CnV register is updated:

**Table 23-11. CnV register update**

When	Then CnV register is updated
CLKS[1:0] = 0:0	When CnV register is written, independent of FTMEN bit.
<ul style="list-style-type: none"> <li>• CLKS[1:0] ≠ 0:0, and</li> <li>• FTMEN = 0</li> </ul>	According to the selected mode, that is:

*Table continues on the next page...*

**Table 23-11. CnV register update (continued)**

When	Then CnV register is updated
	<ul style="list-style-type: none"> <li>If the selected mode is Output Compare, then CnV register is updated on the next FTM counter change, end of the prescaler counting, after CnV register was written.</li> <li>If the selected mode is EPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000.</li> <li>If the selected mode is CPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).</li> </ul>
<ul style="list-style-type: none"> <li>CLKS[1:0] ≠ 0:0, and</li> <li>FTMEN = 1</li> </ul>	<p>According to the selected mode, that is:</p> <ul style="list-style-type: none"> <li>If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter, the end of the prescaler counting. If (SYNCEN = 1) then CnV register is updated by the <a href="#">C(n)V and C(n+1)V register synchronization</a>.</li> <li>If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the <a href="#">C(n)V and C(n+1)V register synchronization</a>.</li> </ul>

## 23.5.11 PWM synchronization

The PWM synchronization provides an opportunity to update the MOD, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

### Note

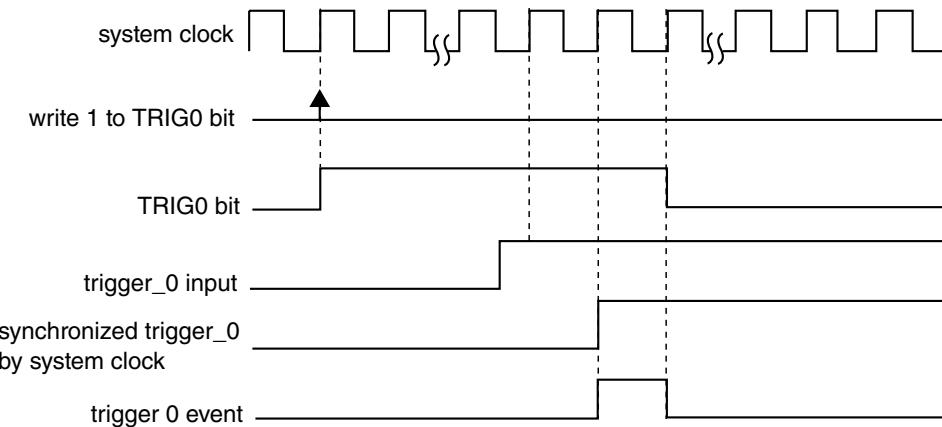
The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, only the enhanced PWM synchronization must be used.

### 23.5.11.1 Hardware trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the system clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGN bit, then the synchronization is initiated, but TRIGN bit remains set due to the write operation.

**Note**

All hardware trigger inputs have the same behavior.

**Figure 23-42. Hardware trigger event with HWTRIGMODE = 0**

If HWTRIGMODE = 1, then the TRIGN bit is only cleared when 0 is written to it.

### NOTE

The HWTRIGMODE bit must be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

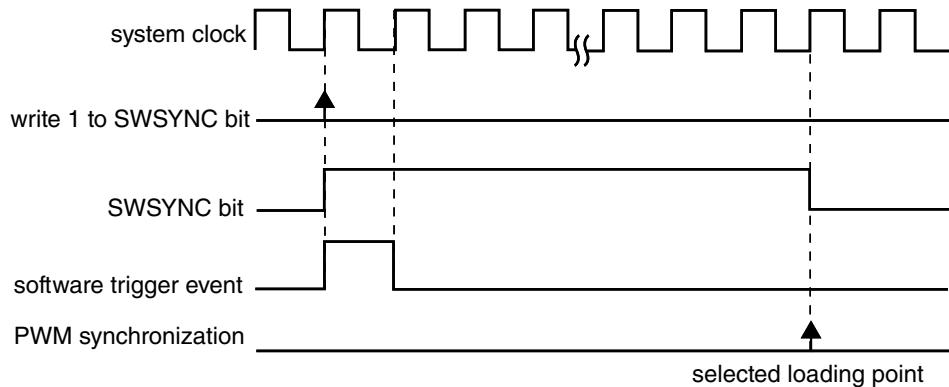
#### 23.5.11.2 Software trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see [Boundary cycle and loading points](#) and the following figure. If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see the following figure. If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.



**Figure 23-43. Software trigger event**

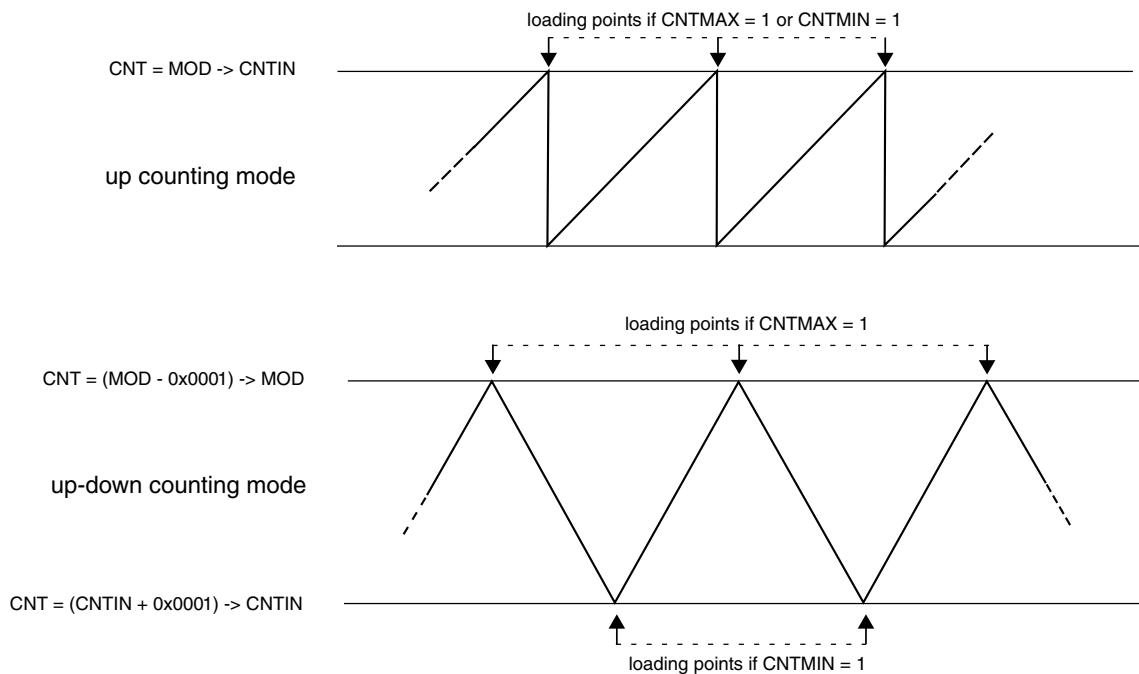
### 23.5.11.3 Boundary cycle and loading points

The boundary cycle definition is important for the loading points for the registers MOD, CNTIN, and C(n)V.

In [Up counting](#) mode, the boundary cycle is defined as when the counter wraps to its initial value (CNTIN). If in [Up-down counting](#) mode, then the boundary cycle is defined as when the counter turns from down to up counting and when from up to down counting.

The following figure shows the boundary cycles and the loading points for the registers. In the Up Counting mode, the loading points are enabled if one of CNTMIN or CTMAX bits are 1. In the Up-Down Counting mode, the loading points are selected by CNTMIN and CNTMAX bits, as indicated in the figure. These loading points are safe places for register updates thus allowing a smooth transitions in PWM waveform generation.

For both counting modes, if neither CNTMIN nor CNTMAX are 1, then the boundary cycles are not used as loading points for registers updates. See the register synchronization descriptions in the following sections for details.



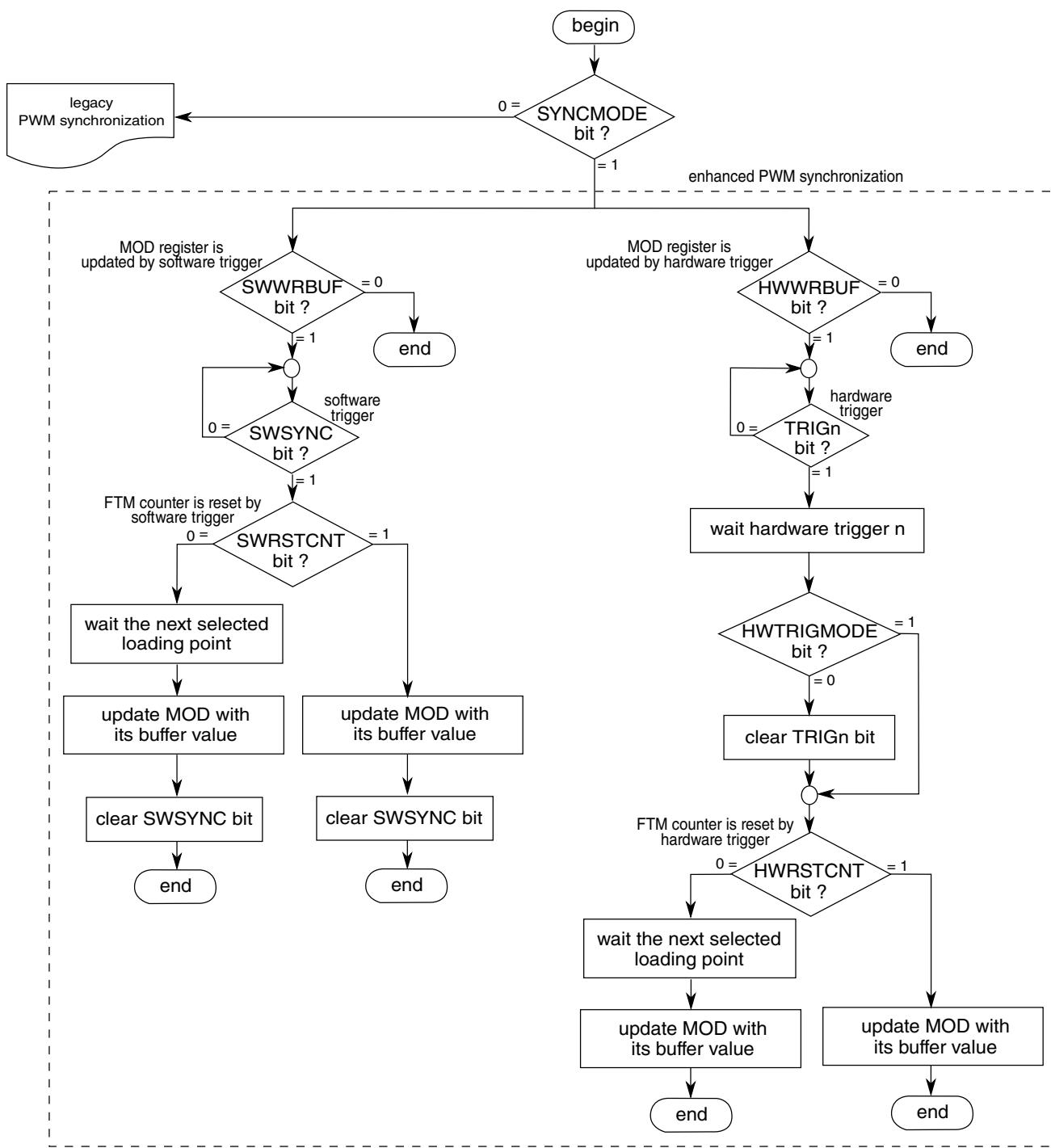
**Figure 23-44. Boundary cycles and loading points**

#### 23.5.11.4 MOD register synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF, and HWRSTCNT bits according to this flowchart:

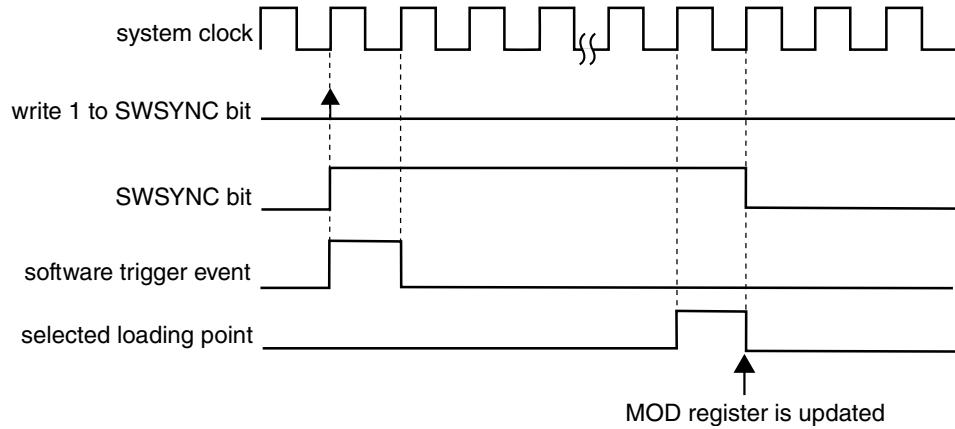


**Figure 23-45. MOD register synchronization flowchart**

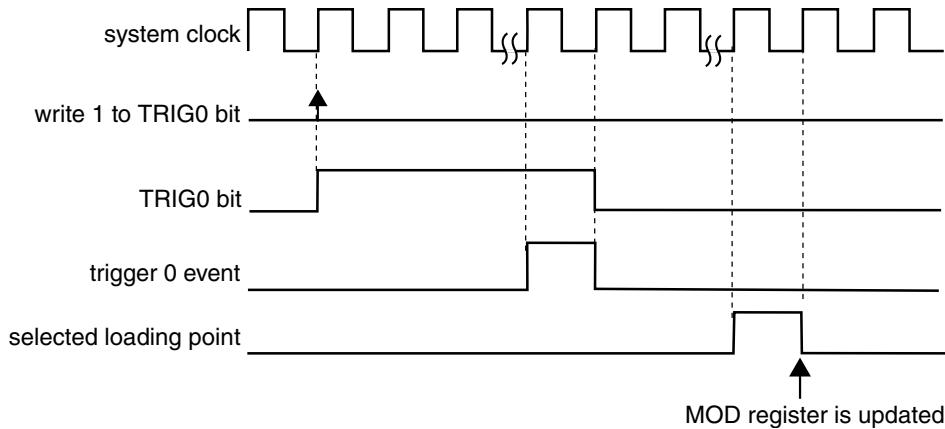
In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 0), then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected

loading point. If the trigger event was a hardware trigger, then the trigger enable bit (TRIGn) is cleared according to [Hardware trigger](#). Examples with software and hardware triggers follow.

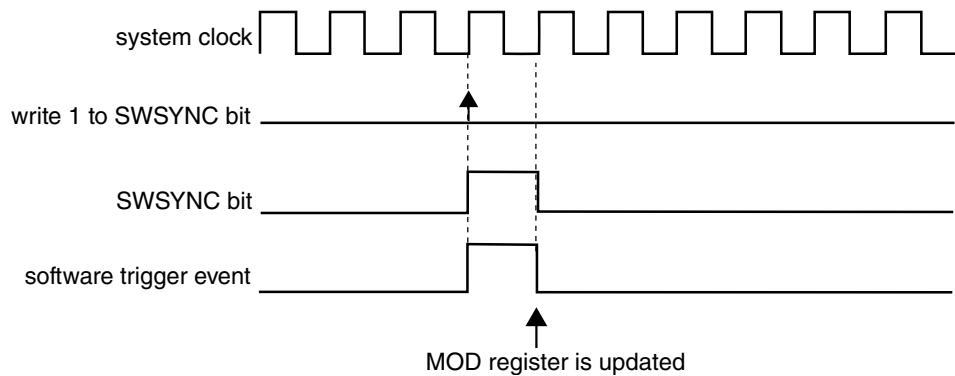


**Figure 23-46. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and software trigger was used**

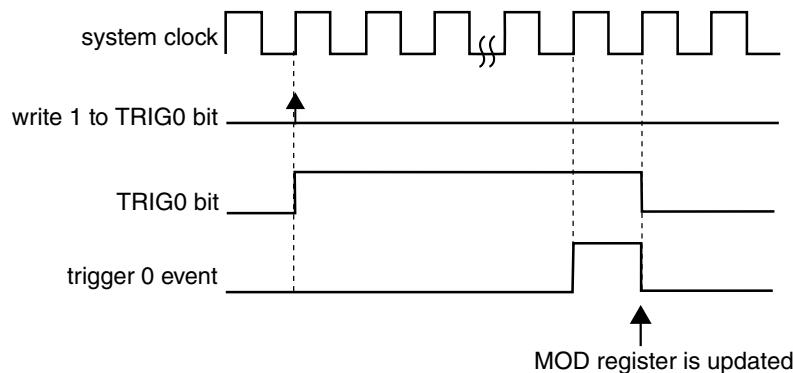


**Figure 23-47. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and a hardware trigger was used**

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 1), then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to [Hardware trigger](#). Examples with software and hardware triggers follow.

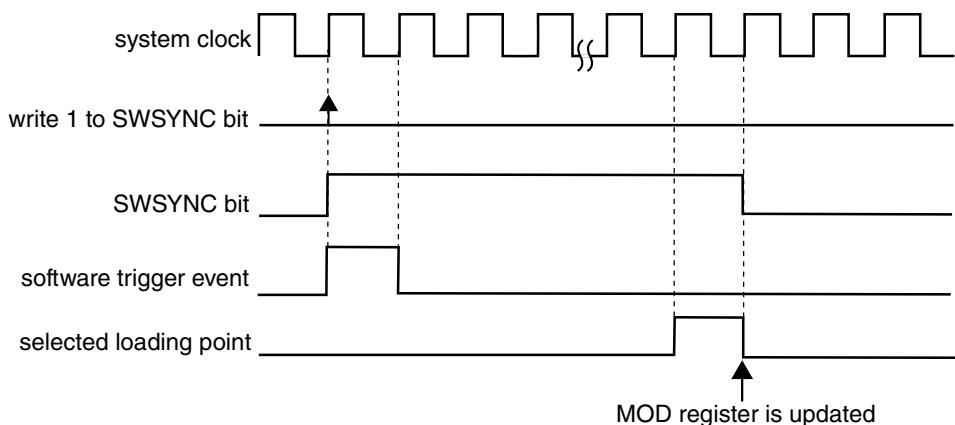


**Figure 23-48. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and software trigger was used**



**Figure 23-49. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and a hardware trigger was used**

If (SYNCMODE = 0) and (PWMSYNC = 1), then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:



**Figure 23-50. MOD synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)**

### 23.5.11.5 CNTIN register synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1), and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization; see [MOD register synchronization](#).

### 23.5.11.6 C(n)V and C(n+1)V register synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

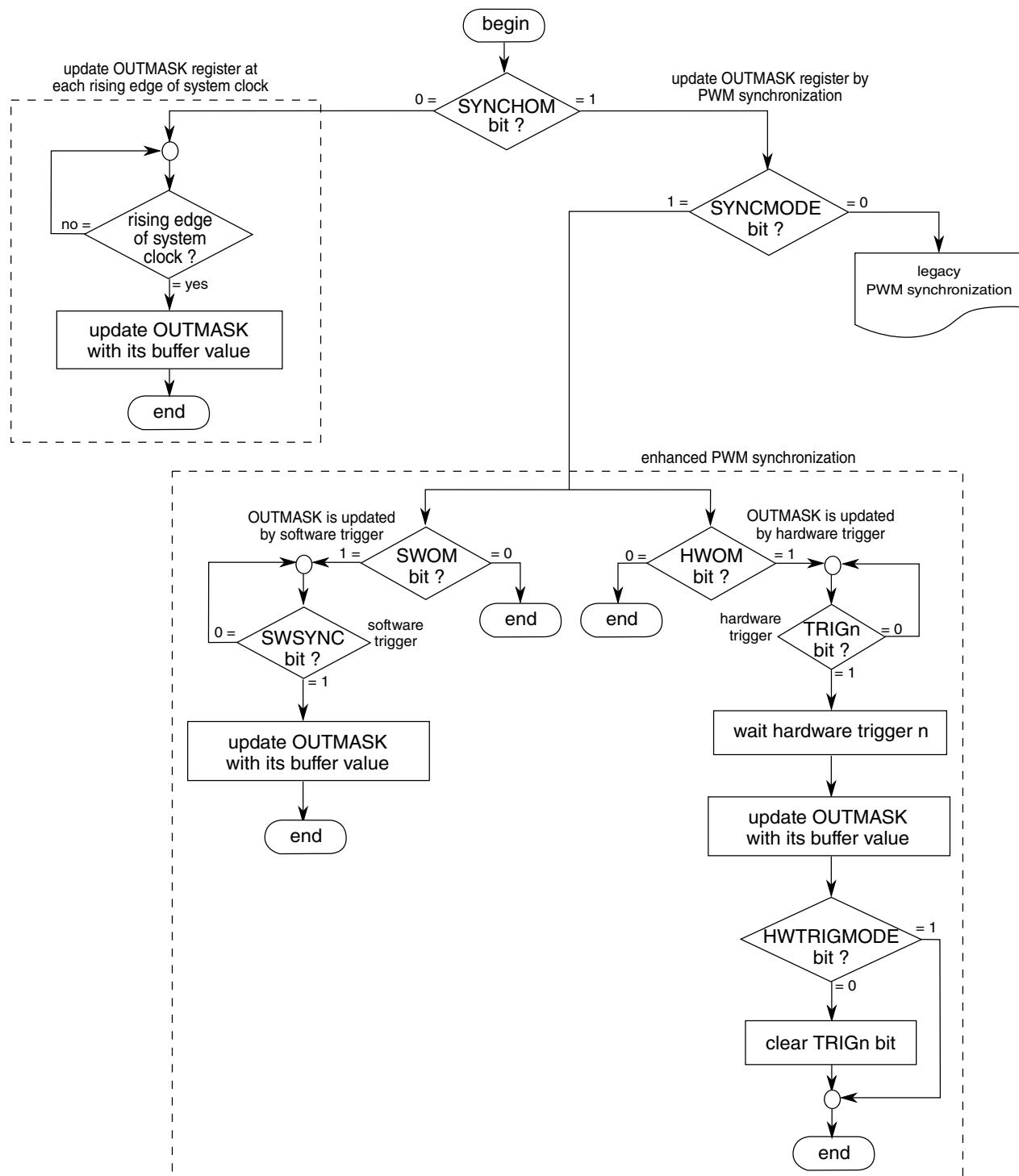
This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the [MOD register synchronization](#). However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

### 23.5.11.7 OUTMASK register synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of system clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

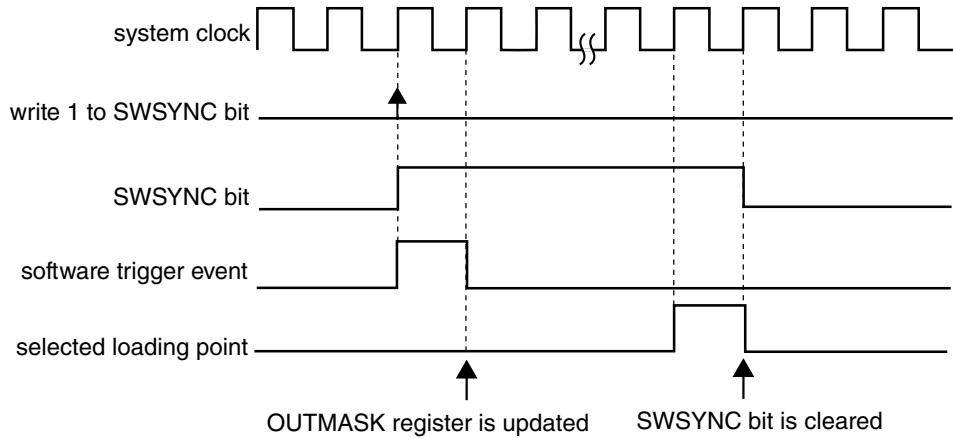


**Figure 23-51. OUTMASK register synchronization flowchart**

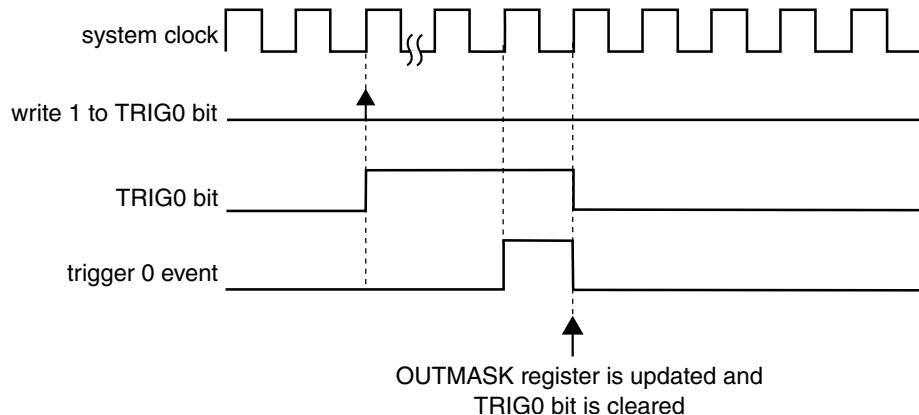
In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

## Functional description

If (**SYNCMODE** = 0), (**SYNCHOM** = 1), and (**PWMSYNC** = 0), then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger, then the **SWSYNC** bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the **TRIGn** bit is cleared according to [Hardware trigger](#). Examples with software and hardware triggers follow.

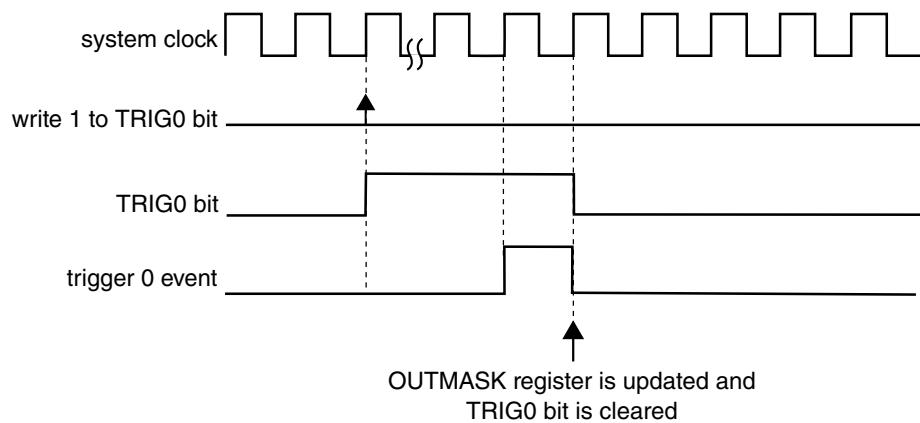


**Figure 23-52. OUTMASK synchronization with (**SYNCMODE** = 0), (**SYNCHOM** = 1), (**PWMSYNC** = 0) and software trigger was used**



**Figure 23-53. OUTMASK synchronization with (**SYNCMODE** = 0), (**HWTRIGMODE** = 0), (**SYNCHOM** = 1), (**PWMSYNC** = 0), and a hardware trigger was used**

If (**SYNCMODE** = 0), (**SYNCHOM** = 1), and (**PWMSYNC** = 1), then this synchronization is made on the next enabled hardware trigger. The **TRIGn** bit is cleared according to [Hardware trigger](#). An example with a hardware trigger follows.



**Figure 23-54. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and a hardware trigger was used**

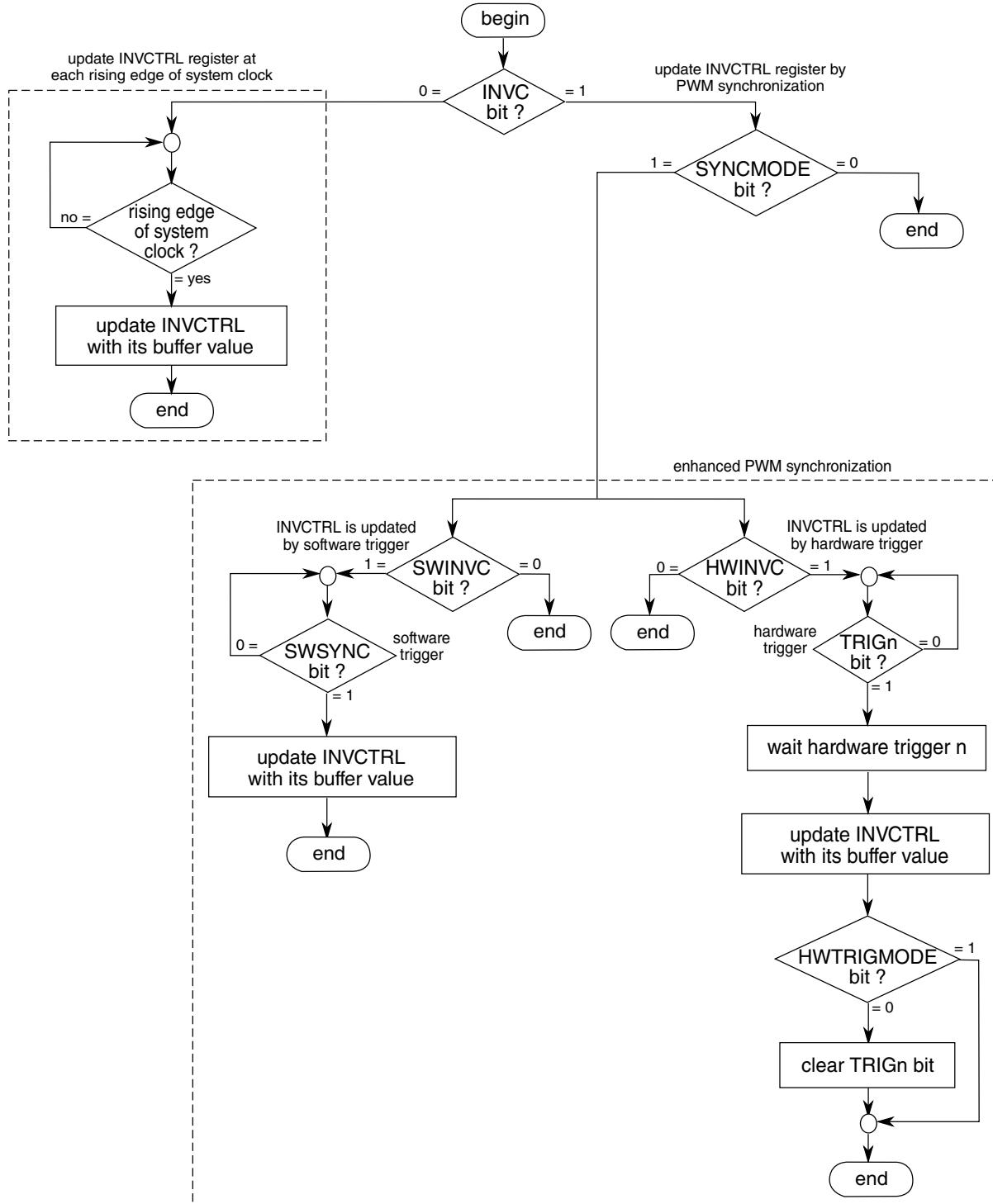
### 23.5.11.8 INVCTRL register synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of system clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

## Functional description



**Figure 23-55. INVCTRL register synchronization flowchart**

### 23.5.11.9 SWOCTRL register synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of system clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

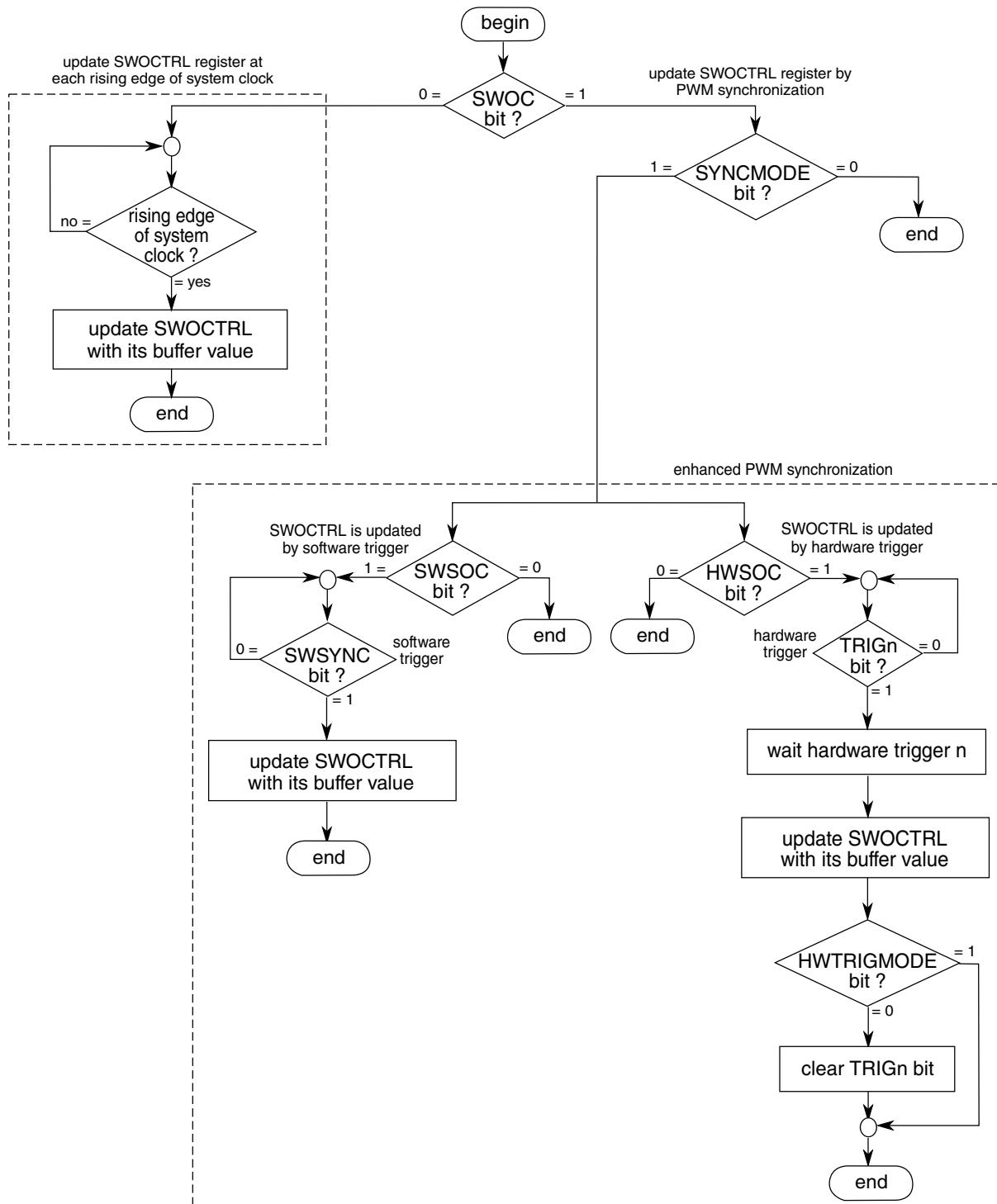
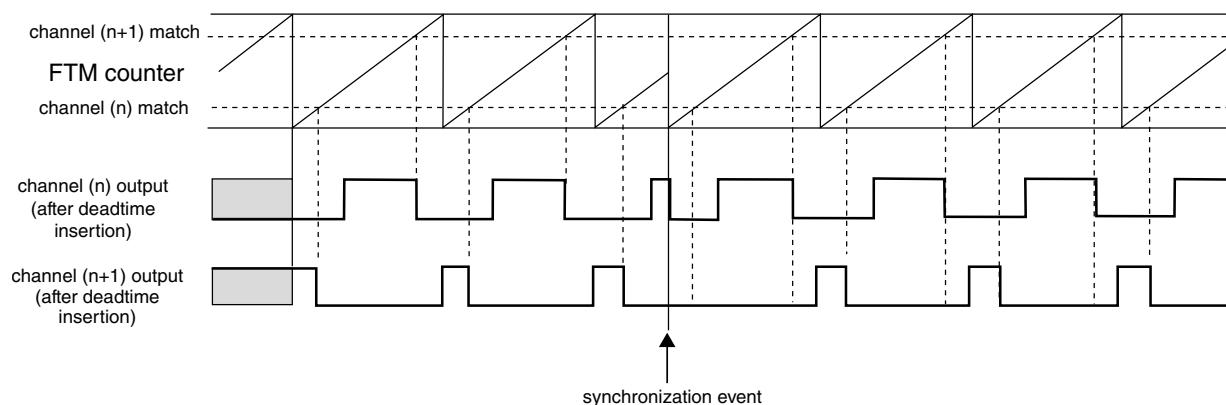


Figure 23-56. SWOCTRL register synchronization flowchart

### 23.5.11.10 FTM counter synchronization

The FTM counter synchronization is a mechanism that allows the FTM to restart the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value, except for channels in Output Compare mode, and the FTM counter is forced to its initial counting value defined by CNTIN register.

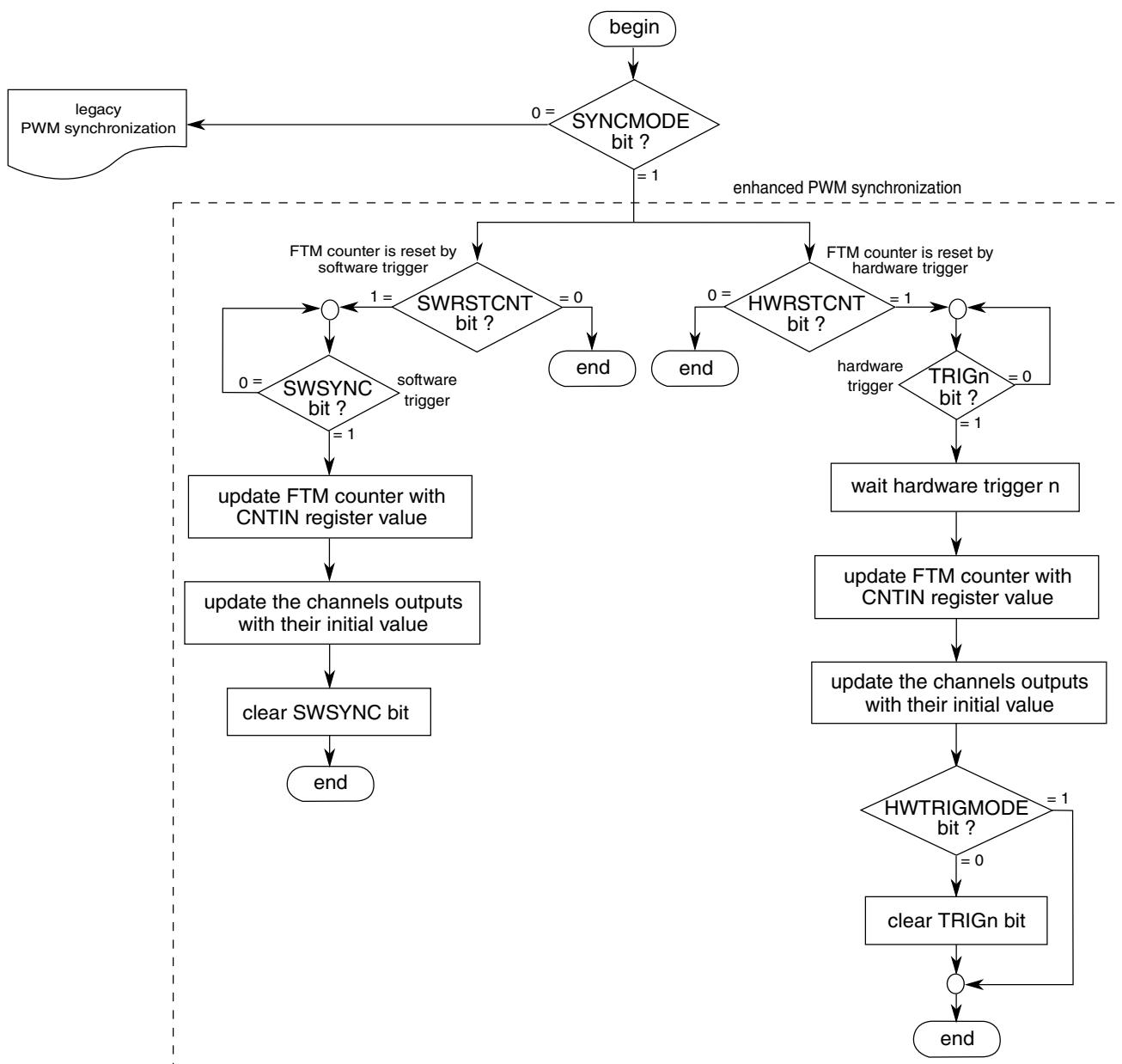
The following figure shows the FTM counter synchronization. Note that after the synchronization event occurs, the channel (n) is set to its initial value and the channel (n+1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected, then the channel (n+1) transitions to logical value 1 immediately after the synchronization event occurs.



**Figure 23-57. FTM counter synchronization**

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, the FTM counter must be synchronized only by the enhanced PWM synchronization.

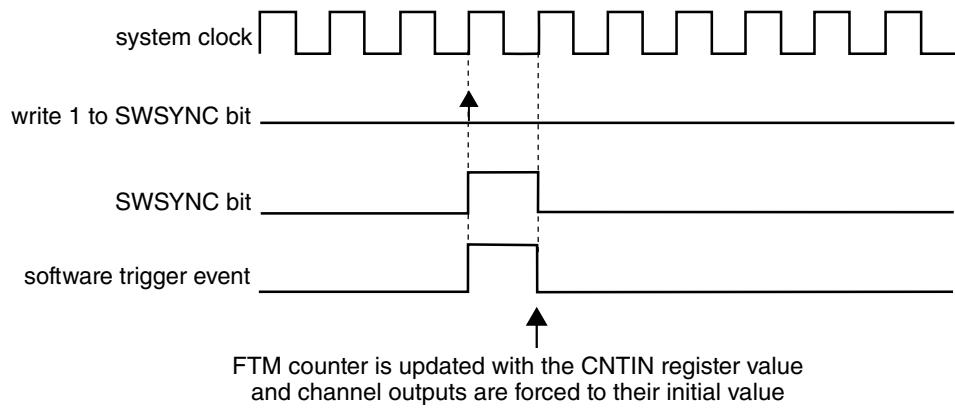
In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

**Figure 23-58. FTM counter synchronization flowchart**

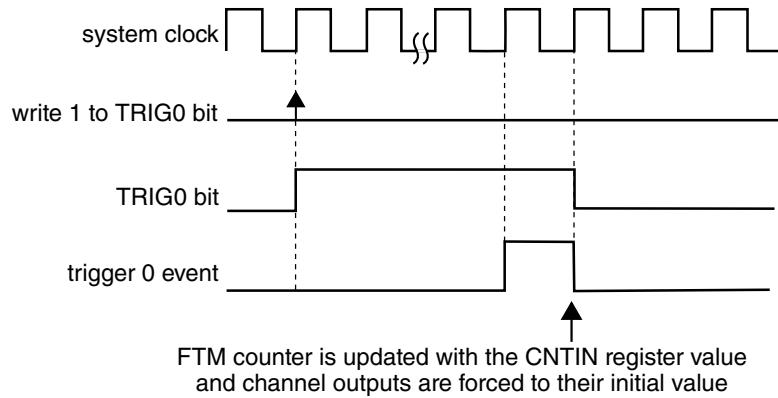
In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to [Hardware trigger](#). Examples with software and hardware triggers follow.

## Functional description

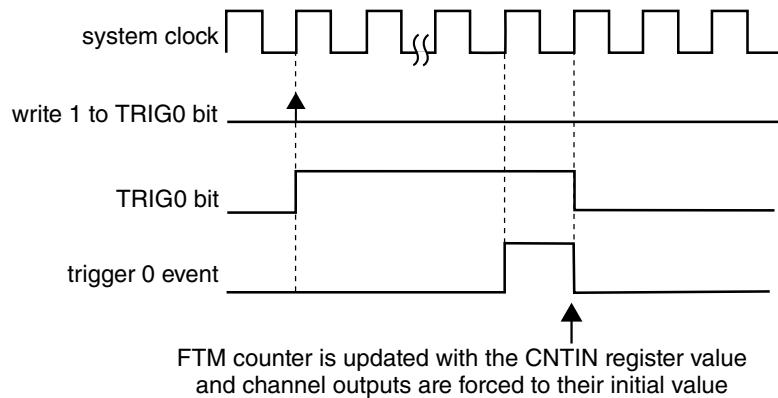


**Figure 23-59. FTM counter synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and software trigger was used**



**Figure 23-60. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and a hardware trigger was used**

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to [Hardware trigger](#).



**Figure 23-61. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and a hardware trigger was used**

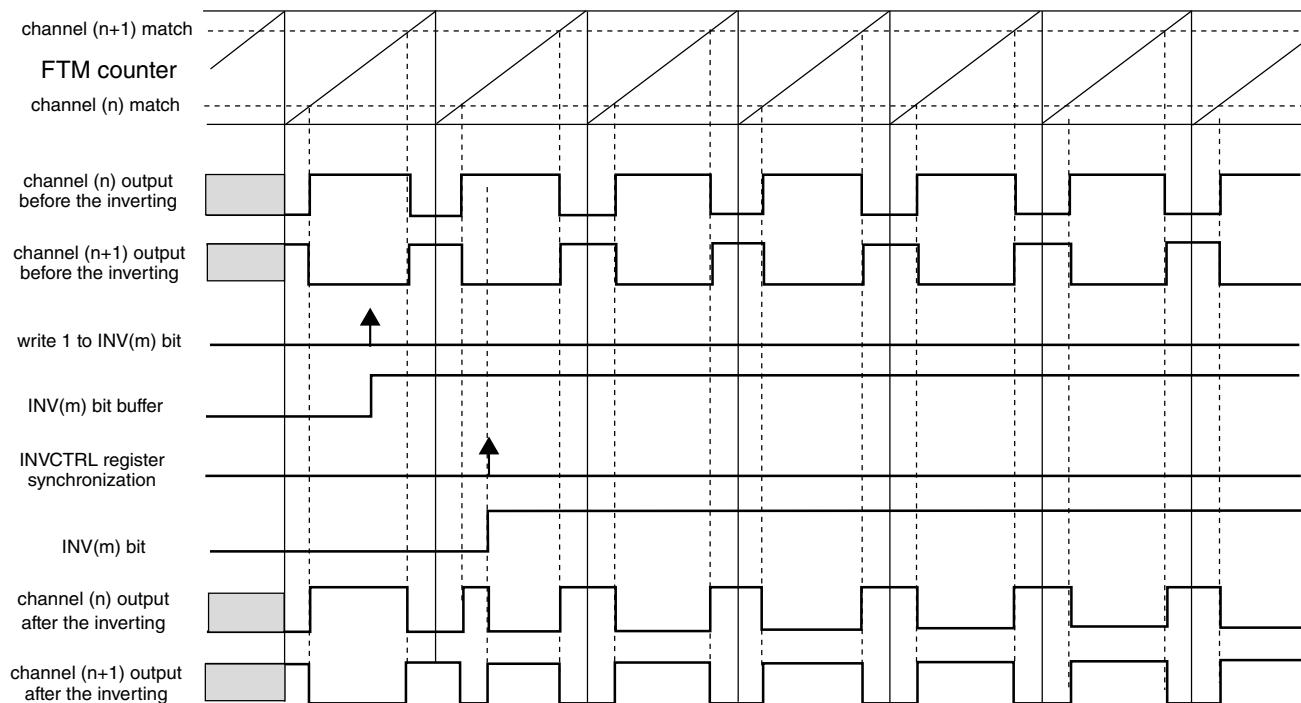
### 23.5.12 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1, and
- INV<sub>m</sub> = 1 (where m represents a channel pair)

The INV<sub>m</sub> bit in INVCTRL register is updated with its buffer value according to [INVCTRL register synchronization](#)

In High-True (ELSnB:ELSnA = 1:0) Combine mode, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.

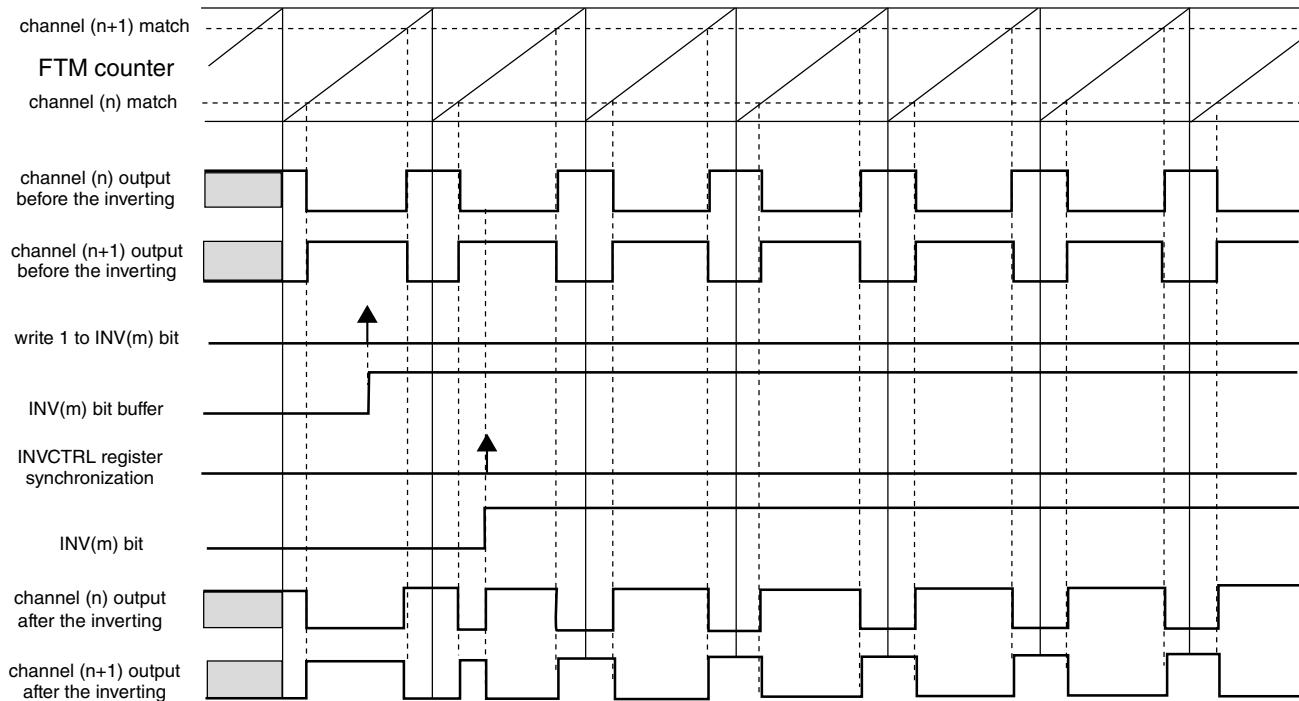


NOTE

INV(m) bit selects the inverting to the pair channels (n) and (n+1).

**Figure 23-62. Channels (n) and (n+1) outputs after the inverting in High-True (ELSnB:ELSnA = 1:0) Combine mode**

Note that the ELSnB:ELSnA bits value should be considered because they define the active state of the channels outputs. In Low-True (ELSnB:ELSnA = X:1) Combine mode, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. When inverting is selected, the channels (n) and (n+1) present waveforms as shown in the following figure.



## NOTE

INV(m) bit selects the inverting to the pair channels (n) and (n+1).

**Figure 23-63. Channels (n) and (n+1) outputs after the inverting in Low-True (ELSnB:ELSnA = X:1) Combine mode**

### Note

The inverting feature is not available in Output Compare mode.

### 23.5.13 Software output control

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

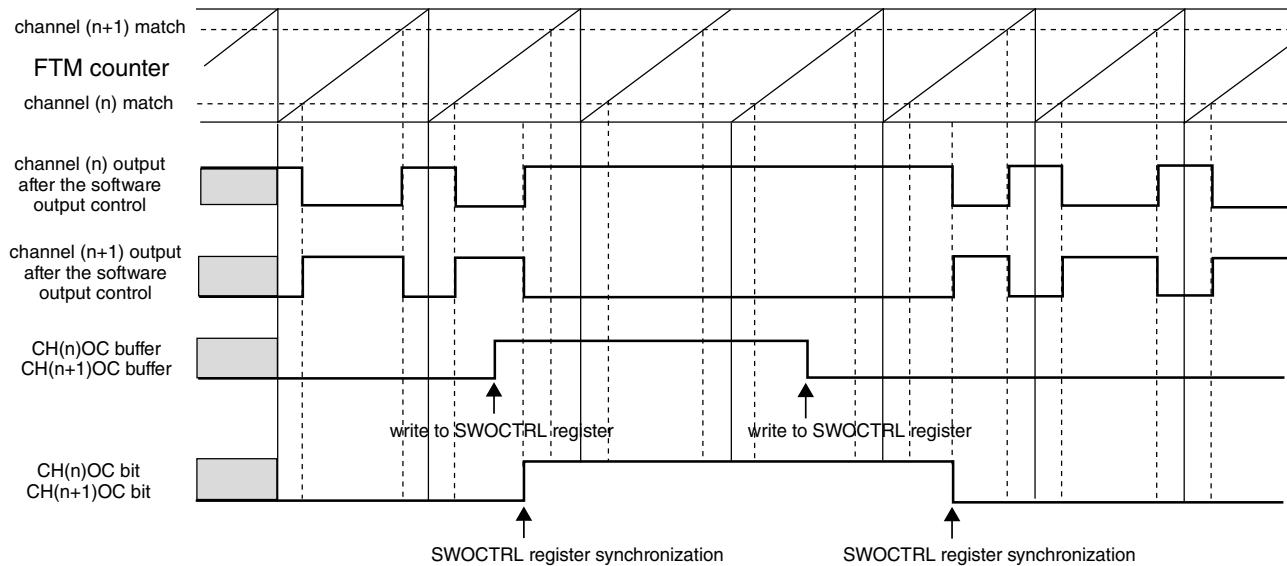
The software output control is selected when:

- QUADEN = 0
- DECAPEN = 0, and
- CHnOC = 1

The CHnOC bit enables the software output control for a specific channel output and the CHnOCV selects the value that is forced to this channel output.

Both CHnOC and CHnOCV bits in SWOCTRL register are buffered and updated with their buffer value according to [SWOCTRL register synchronization](#).

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to Combine and Complementary mode.



**NOTE**

CH(n)OCV = 1 and CH(n+1)OCV = 0.

**Figure 23-64. Example of software output control in Combine and Complementary mode**

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

**Table 23-12. Software output control behavior when (COMP = 0)**

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

**Table 23-13. Software output control behavior when (COMP = 1)**

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

**Note**

- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit must not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n+1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS field description in the Status and Control register).

### 23.5.14 Deadtime insertion

The deadtime insertion is enabled when (DTEN = 1) and (DTVAL[5:0] is non-zero).

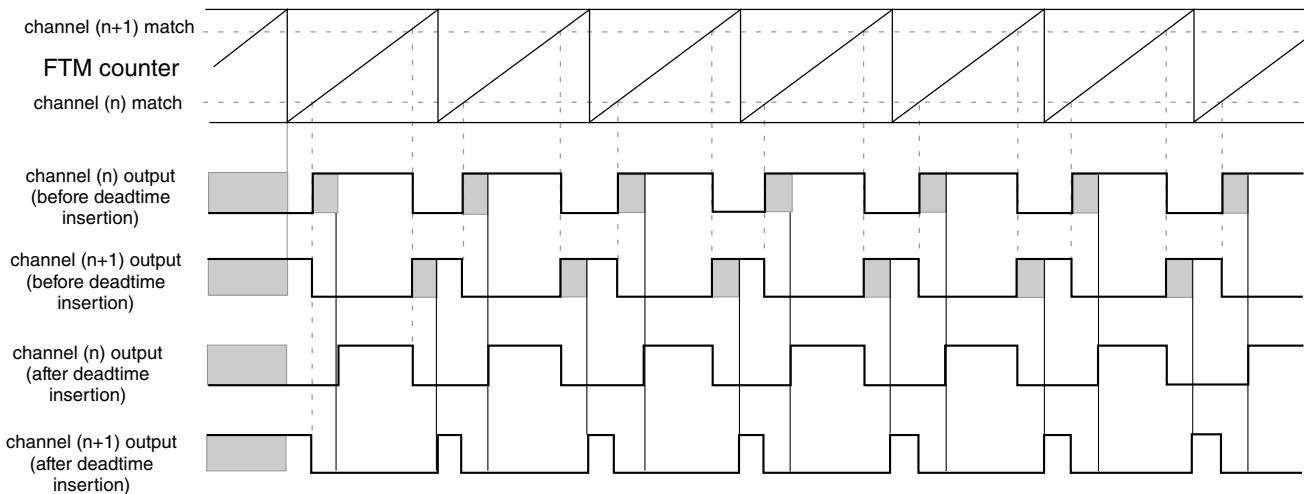
DEADTIME register defines the deadtime delay that can be used for all FTM channels. The DTAPS[1:0] bits define the prescaler for the system clock and the DTVAL[5:0] bits define the deadtime modulo, that is, the number of the deadtime prescaler clocks.

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

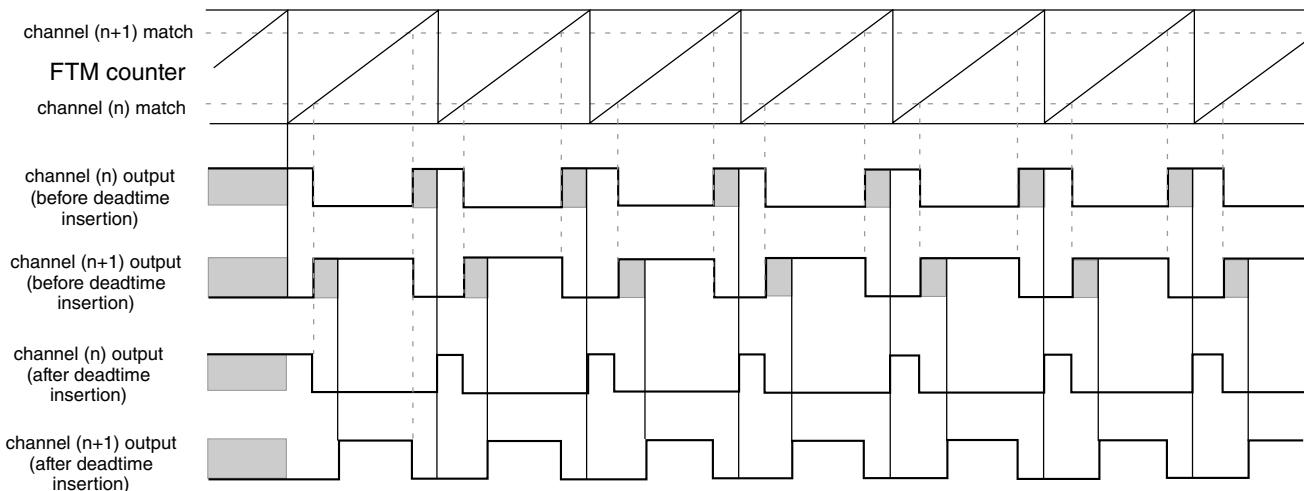
If POL(n) = 0, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If POL(n) = 1, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly,

when the channel (n+1) match (FTM counter =  $C(n+1)V$ ) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n+1) output is cleared.



**Figure 23-65. Deadtime insertion with  $ELSnB:ELSnA = 1:0$ ,  $POL(n) = 0$ , and  $POL(n+1) = 0$**



**Figure 23-66. Deadtime insertion with  $ELSnB:ELSnA = X:1$ ,  $POL(n) = 0$ , and  $POL(n+1) = 0$**

### NOTE

- The deadtime feature must be used only in Complementary mode.
- The deadtime feature is not available in Output Compare mode.

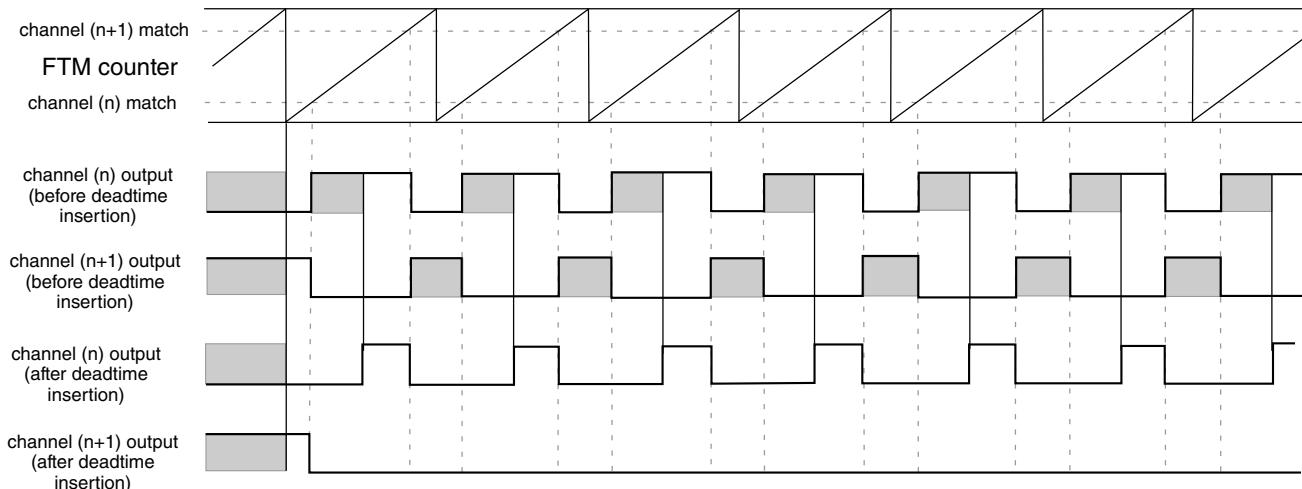
#### 23.5.14.1 Deadtime insertion corner cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

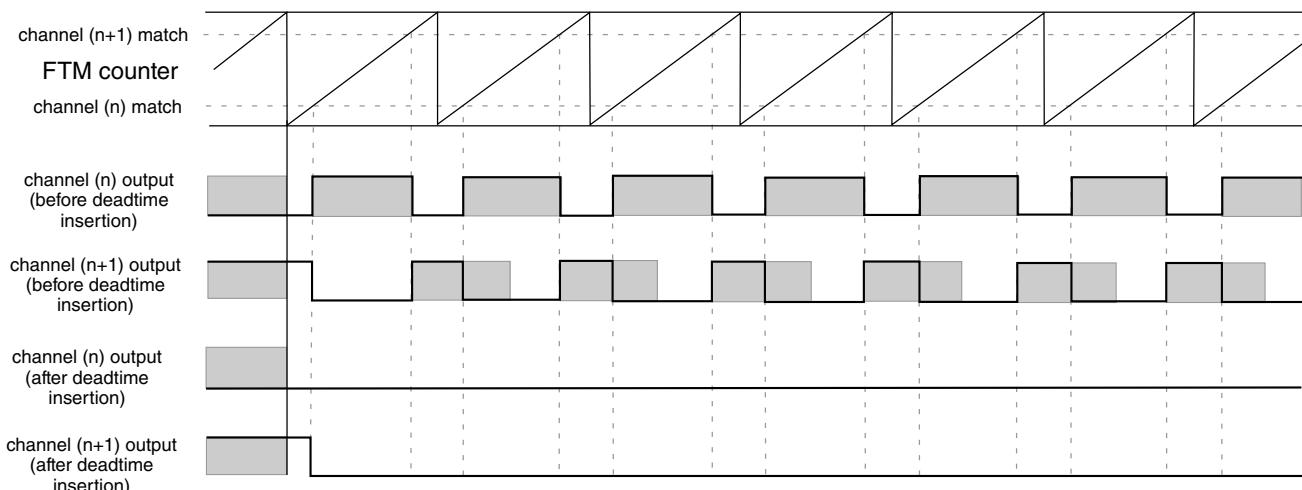
## Functional description

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ( $(C(n) + 1)V - C(n)V) \times \text{system clock}$ ), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ( $(MOD - CNTIN + 1 - (C(n+1)V - C(n)V)) \times \text{system clock}$ ), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.



**Figure 23-67. Example of the deadtime insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channel (n+1) duty cycle**



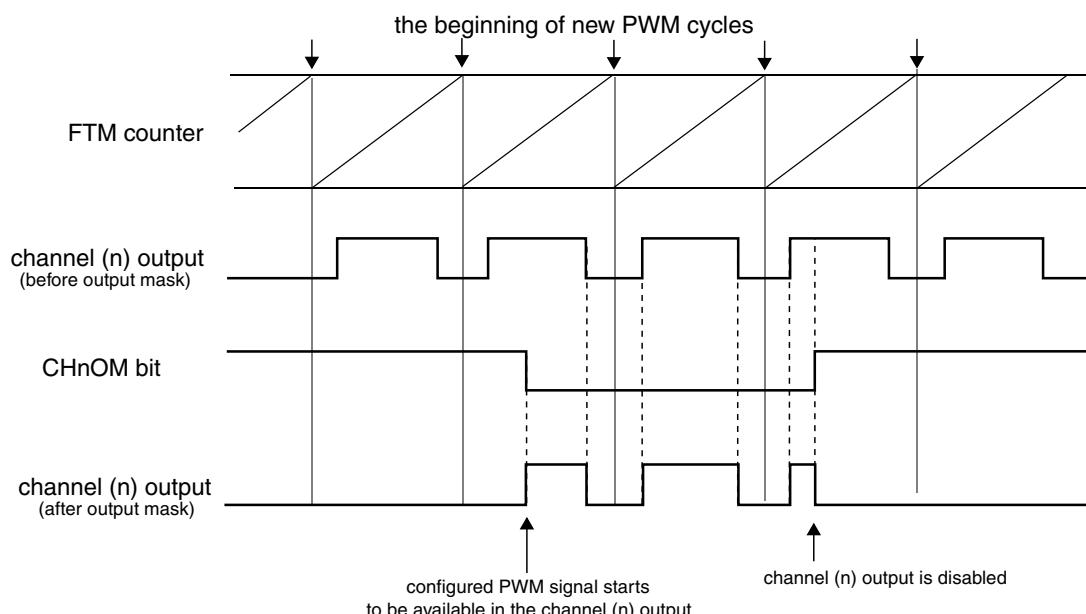
**Figure 23-68. Example of the deadtime insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channels (n) and (n+1) duty cycle**

### 23.5.15 Output mask

The output mask can be used to force channels output to their inactive state through software. For example: to control a BLDC motor.

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization; see [OUTMASK register synchronization](#).

If CHnOM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CHnOM = 0, then the channel (n) output is unaffected by the output mask. See the following figure.



**Figure 23-69. Output mask with POLn = 0**

The following table shows the output mask result before the polarity control.

**Table 23-14. Output mask result for channel (n) before the polarity control**

CHnOM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

## 23.5.16 Fault control

The fault control is enabled if ( $\text{FAULTM}[1:0] \neq 0:0$ ).

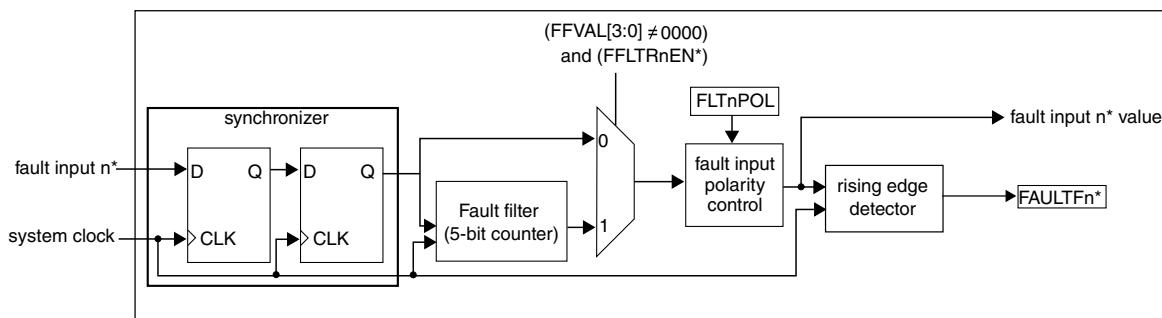
FTM can have up to four fault inputs.  $\text{FAULTTnEN}$  bit (where  $n = 0, 1, 2, 3$ ) enables the fault input  $n$  and  $\text{FFLTrnEN}$  bit enables the fault input  $n$  filter.  $\text{FFVAL}[3:0]$  bits select the value of the enabled filter in each enabled fault input.

First, each fault input signal is synchronized by the system clock; see the synchronizer block in the following figure. Following synchronization, the fault input  $n$  signal enters the filter block. When there is a state change in the fault input  $n$  signal, the 5-bit counter is reset and starts counting up. As long as the new state is stable on the fault input  $n$ , the counter continues to increment. If the 5-bit counter overflows, that is, the counter exceeds the value of the  $\text{FFVAL}[3:0]$  bits, the new fault input  $n$  value is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the fault input  $n$  signal before validation (counter overflow), the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by  $\text{FFVAL}[3:0]$  bits ( $\times$  system clock) is regarded as a glitch and is not passed on to the edge detector.

The fault input  $n$  filter is disabled when the  $\text{FFVAL}[3:0]$  bits are zero or when  $\text{FAULTTnEN} = 0$ . In this case, the fault input  $n$  signal is delayed 2 rising edges of the system clock and the  $\text{FAULTFn}$  bit is set on 3th rising edge of the system clock after a rising edge occurs on the fault input  $n$ .

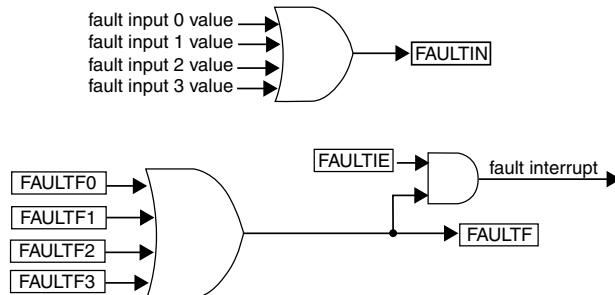
If  $\text{FFVAL}[3:0] \neq 0000$  and  $\text{FAULTTnEN} = 1$ , then the fault input  $n$  signal is delayed ( $3 + \text{FFVAL}[3:0]$ ) rising edges of the system clock, that is, the  $\text{FAULTFn}$  bit is set ( $4 + \text{FFVAL}[3:0]$ ) rising edges of the system clock after a rising edge occurs on the fault input  $n$ .



\* where  $n = 3, 2, 1, 0$

**Figure 23-70. Fault input n control block diagram**

If the fault control and fault input n are enabled and a rising edge at the fault input n signal is detected, a fault condition has occurred and the FAULTFn bit is set. The FAULTF bit is the logic OR of FAULTFn[3:0] bits. See the following figure.



**Figure 23-71. FAULTF and FAULTIN bits and fault interrupt**

If the fault control is enabled ( $\text{FAULTM}[1:0] \neq 0:0$ ), a fault condition has occurred and ( $\text{FAULTEN} = 1$ ), then outputs are forced to their safe values:

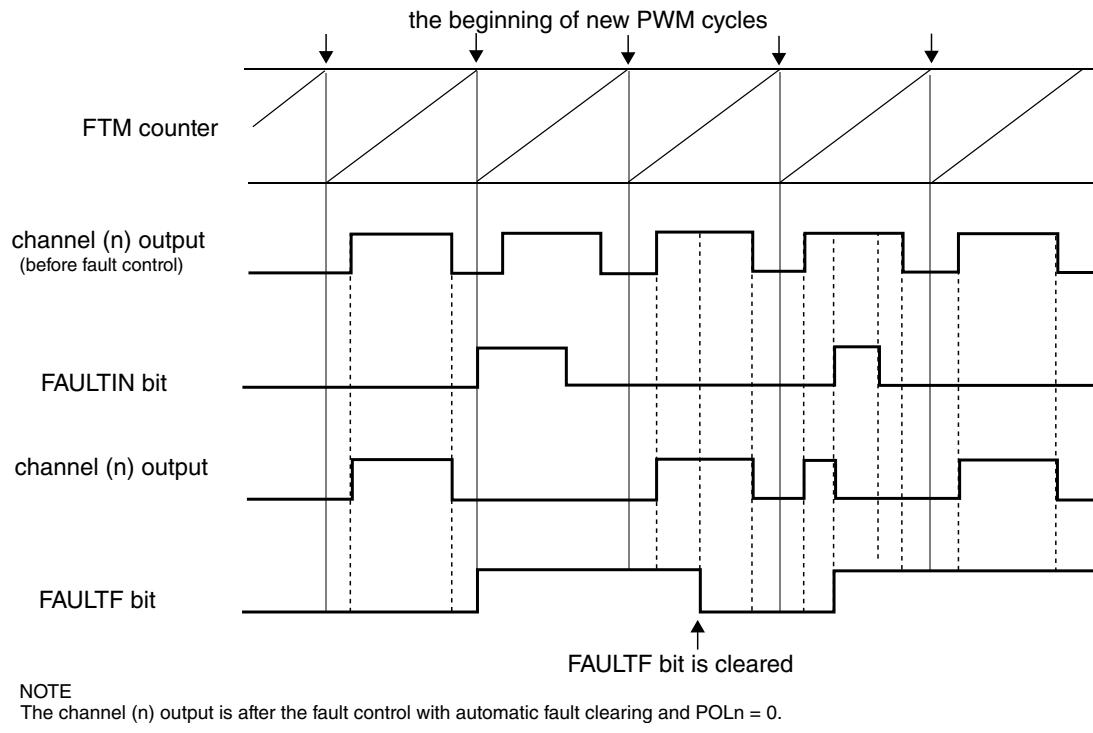
- Channel (n) output takes the value of  $\text{POL}(n)$
- Channel (n+1) takes the value of  $\text{POL}(n+1)$

The fault interrupt is generated when ( $\text{FAULTF} = 1$ ) and ( $\text{FAULTIE} = 1$ ). This interrupt request remains set until:

- Software clears the FAULTF bit by reading FAULTF bit as 1 and writing 0 to it
- Software clears the FAULTIE bit
- A reset occurs

### 23.5.16.1 Automatic fault clearing

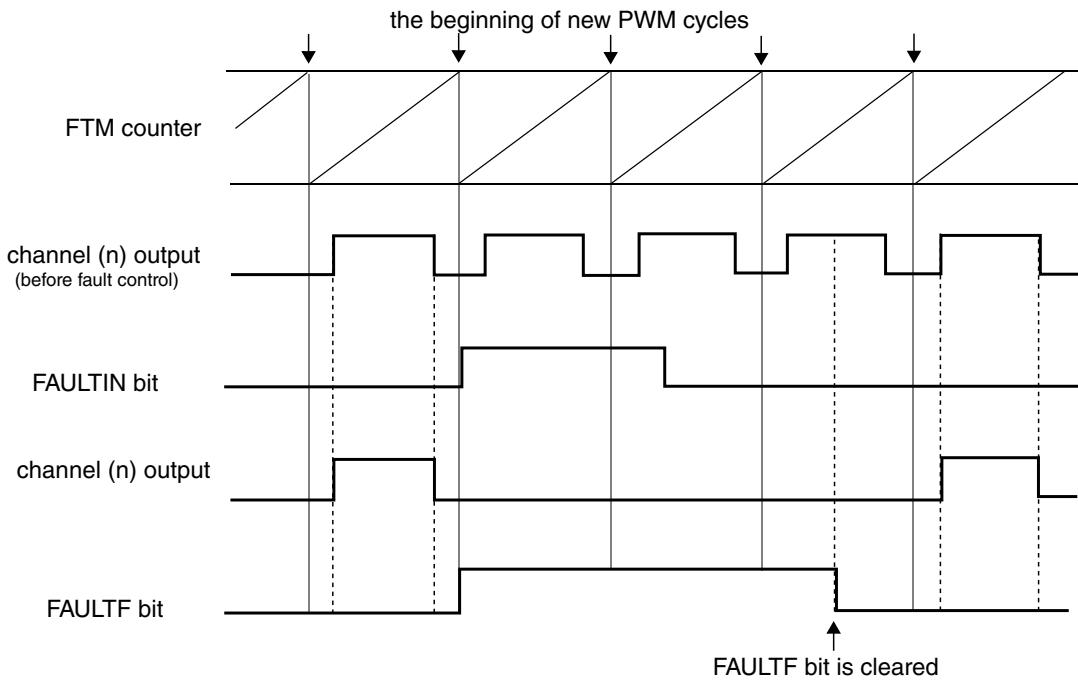
If the automatic fault clearing is selected ( $\text{FAULTM}[1:0] = 1:1$ ), then the channels output disabled by fault control is again enabled when the fault input signal (FAULTIN) returns to zero and a new PWM cycle begins. See the following figure.



**Figure 23-72. Fault control with automatic fault clearing**

### 23.5.16.2 Manual fault clearing

If the manual fault clearing is selected ( $\text{FAULTM}[1:0] = 0:1$  or  $1:0$ ), then the channels output disabled by fault control is again enabled when the FAULTF bit is cleared and a new PWM cycle begins. See the following figure.



**Figure 23-73. Fault control with manual fault clearing**

### 23.5.16.3 Fault inputs polarity control

The FLTjPOL bit selects the fault input j polarity, where  $j = 0, 1, 2, 3$ :

- If  $\text{FLTjPOL} = 0$ , the fault j input polarity is high, so the logical one at the fault input j indicates a fault.
- If  $\text{FLTjPOL} = 1$ , the fault j input polarity is low, so the logical zero at the fault input j indicates a fault.

### 23.5.17 Polarity control

The POLn bit selects the channel (n) output polarity:

- If  $\text{POLn} = 0$ , the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If  $\text{POLn} = 1$ , the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

### 23.5.18 Initialization

The initialization forces the CH<sub>n</sub>OI bit value to the channel (n) output when a one is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

**Table 23-15. Initialization behavior when (COMP = 0 and DTEN = 0)**

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

**Table 23-16. Initialization behavior when (COMP = 1 or DTEN = 1)**

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

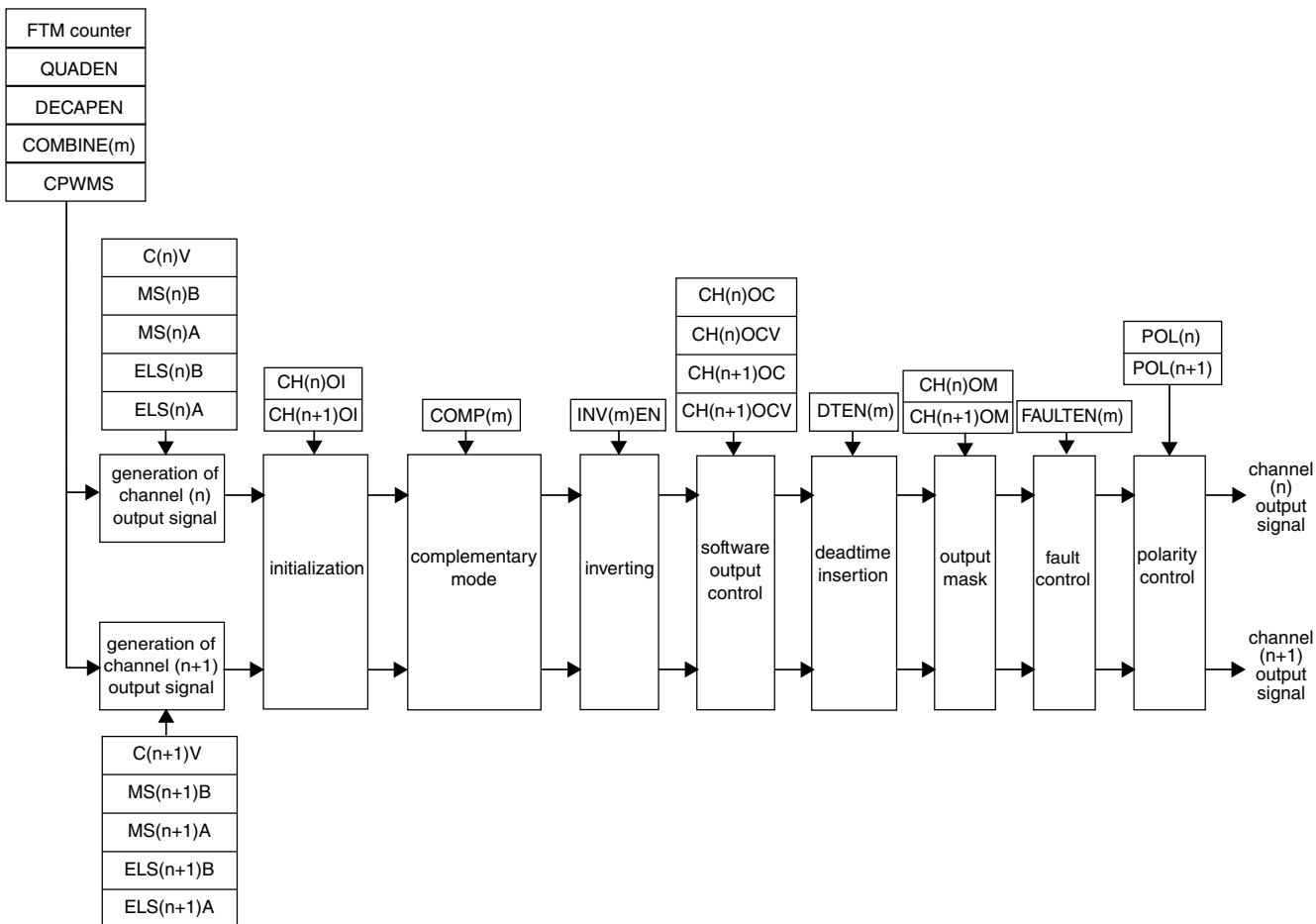
#### Note

The initialization feature must be used only with disabled FTM counter. See the description of the CLKS field in the Status and Control register.

### 23.5.19 Features priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.

pair channels (m) - channels (n) and (n+1)

**NOTE**

The channels (n) and (n+1) are in output compare, EPWM, CPWM or combine modes.

**Figure 23-74. Priority of the features used at the generation of channels (n) and (n+1) outputs signals**

### Note

The **Initialization** feature must not be used with **Inverting** and **Software output control** features.

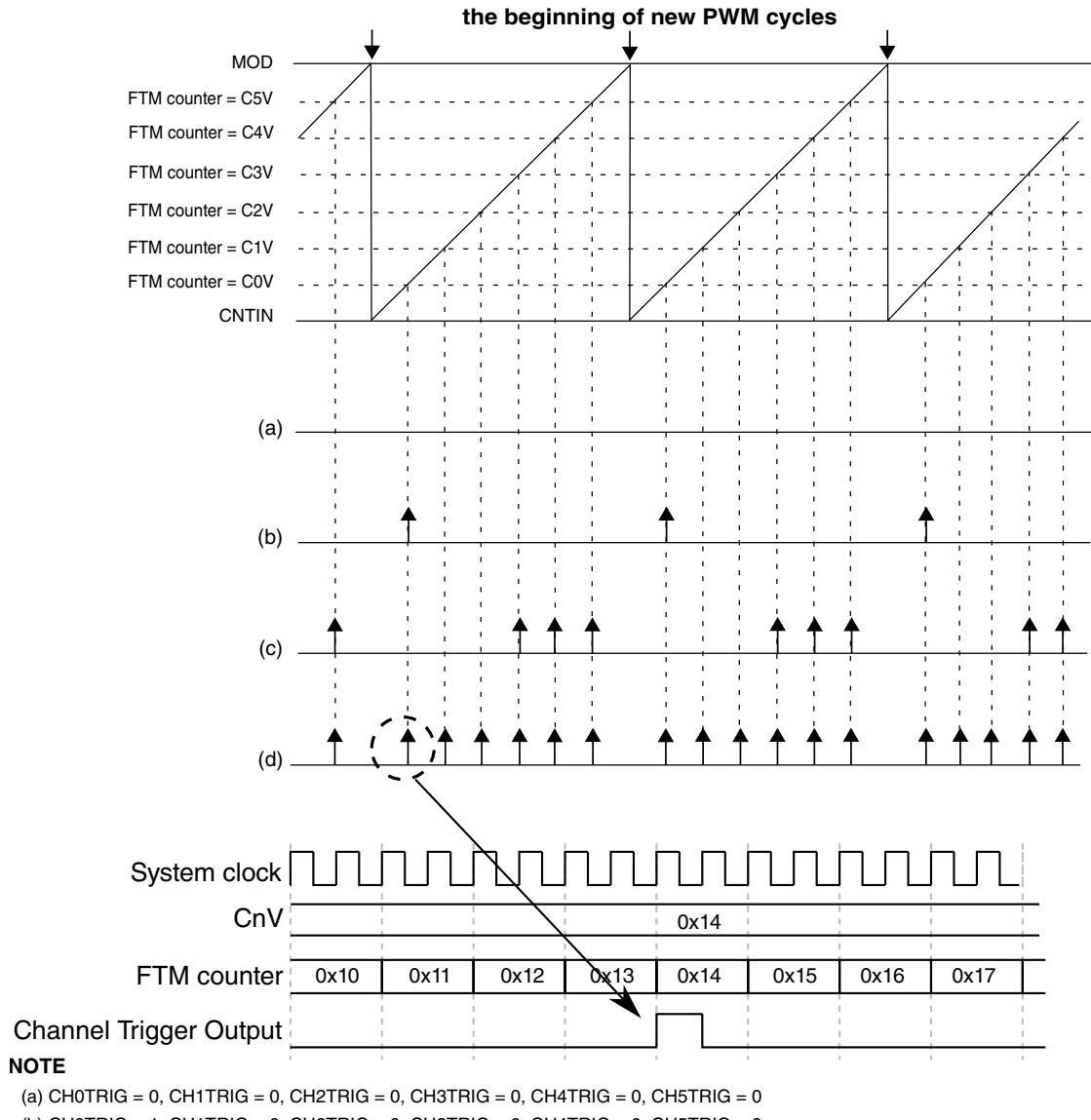
## 23.5.20 Channel trigger output

If CH(j)TRIG bit of the FTM External Trigger (FTM\_EXTTRIG) register is set, where j = 0, 1, 2, 3, 4, or 5, then the FTM generates a trigger when the channel (j) match occurs (FTM counter = C(j)V).

The channel trigger output provides a trigger signal which has one FTM clock period width and is used for on-chip modules.

## Functional description

The FTM is able to generate multiple triggers in one PWM period. Because each trigger is generated for a specific channel, several channels are required to implement this functionality. This behavior is described in the following figure.



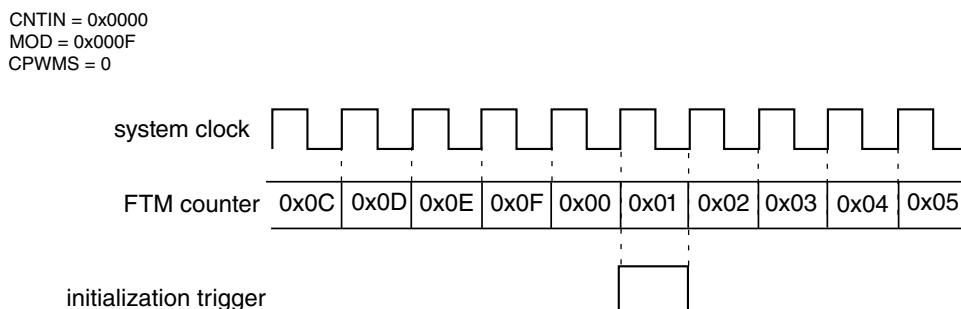
**Figure 23-75. Channel match trigger**

## 23.5.21 Initialization trigger

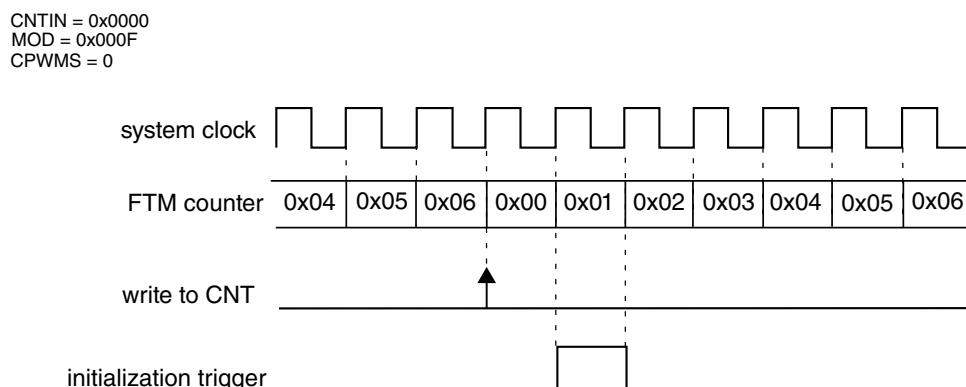
If INITTRIGEN = 1, then the FTM generates a trigger when the FTM counter is updated with the CNTIN register value in the following cases.

- The FTM counter is automatically updated with the CNTIN register value by the selected counting mode.
- When there is a write to CNT register.
- When there is the [FTM counter synchronization](#).
- If ( $CNT = CNTIN$ ), ( $CLKS[1:0] = 0:0$ ), and a value different from zero is written to  $CLKS[1:0]$  bits.

The following figures show these cases.



**Figure 23-76. Initialization trigger is generated when the FTM counting achieves the CNTIN register value**



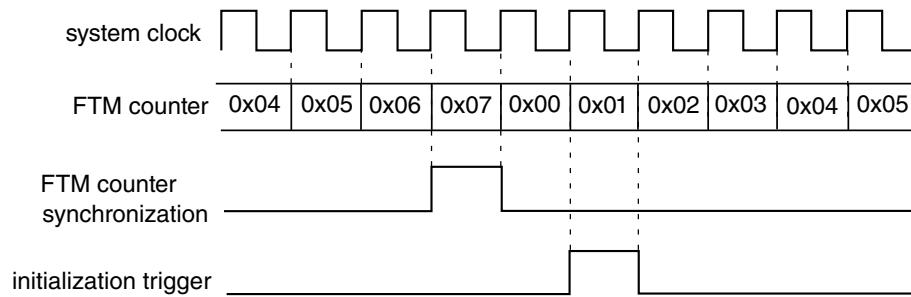
**Figure 23-77. Initialization trigger is generated when there is a write to CNT register**

#### NOTE

The behavior depicted in the [Figure 23-77](#) is not available on CPWM.

## Functional description

CNTIN = 0x0000  
MOD = 0x000F  
CPWMS = 0

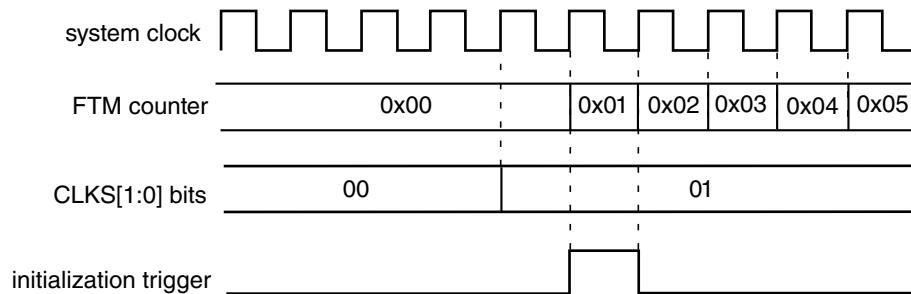


**Figure 23-78. Initialization trigger is generated when there is the FTM counter synchronization**

### NOTE

The behavior depicted in the [Figure 23-78](#) is not available on CPWM.

CNTIN = 0x0000  
MOD = 0x000F  
CPWMS = 0



**Figure 23-79. Initialization trigger is generated if (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits**

### NOTE

The behavior depicted in the [Figure 23-79](#) is not available on CPWM.

The initialization trigger output provides a trigger signal that is used for on-chip modules.

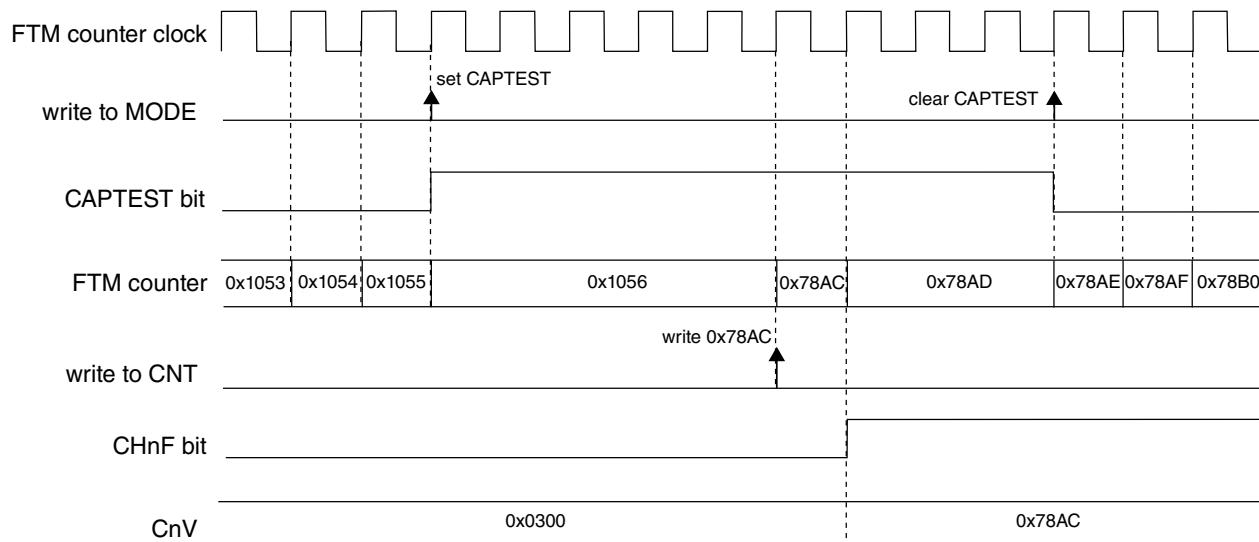
## 23.5.22 Capture Test mode

The Capture Test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for [Input Capture mode](#) and FTM counter must be configured to the [Up counting](#).

When the Capture Test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter; see the following figure. After it was written, all CnV registers are updated with the written value to CNT register and CHnF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration. Its next value depends on CNTIN, MOD, and the written value to FTM counter.

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.



#### NOTE

- FTM counter configuration: (FTMEN = 1), (QUADEN = 0), (CAPTEST = 1), (CPWMS = 0), (CNTIN = 0x0000), and (MOD = 0xFFFF)
- FTM channel n configuration: input capture mode - (DECAPEN = 0), (COMBINE = 0), and (MSnB:MSnA = 0:0)

**Figure 23-80. Capture Test mode**

### 23.5.23 DMA

The channel generates a DMA transfer request according to DMA and CHnIE bits. See the following table.

**Table 23-17. Channel DMA transfer request**

DMA	CHnIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHnF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.

*Table continues on the next page...*

**Table 23-17. Channel DMA transfer request (continued)**

DMA	CHnIE	Channel DMA Transfer Request	Channel Interrupt
1	1	The channel DMA transfer request is generated if (CHnF = 1).	The channel interrupt is not generated.

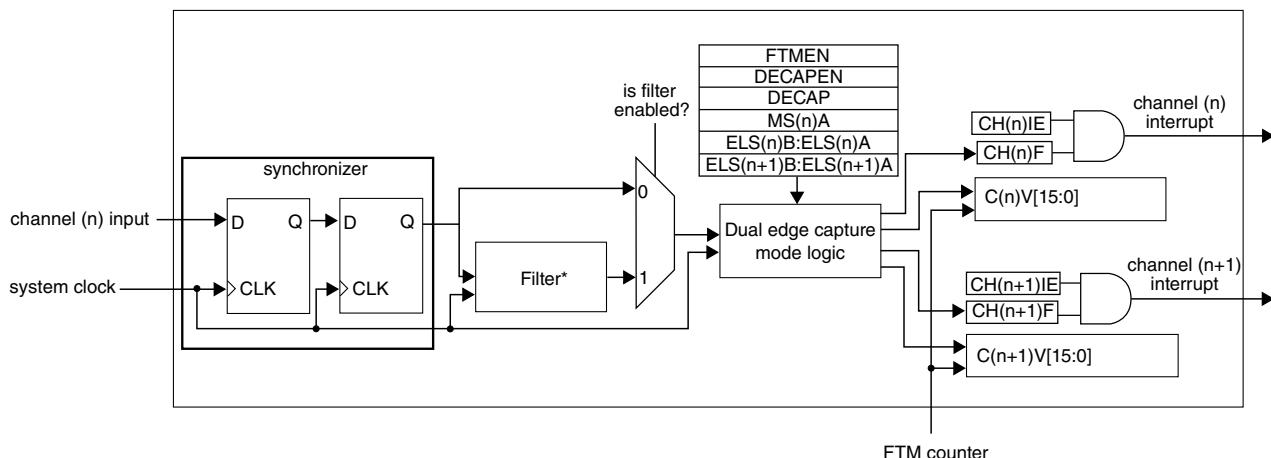
If DMA = 1, the CHnF bit is cleared either by channel DMA transfer done or reading CnSC while CHnF is set and then writing a zero to CHnF bit according to CHnIE bit. See the following table.

**Table 23-18. Clear CHnF bit when DMA = 1**

CHnIE	How CHnF Bit Can Be Cleared
0	CHnF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHnF is set and then writing a 0 to CHnF bit.
1	CHnF bit is cleared when the channel DMA transfer is done.

### 23.5.24 Dual Edge Capture mode

The Dual Edge Capture mode is selected if DECAPEN = 1. This mode allows to measure a pulse width or period of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode when n is 0 or 2.



\* Filtering function for dual edge capture mode is only available in the channels 0 and 2

**Figure 23-81. Dual Edge Capture mode block diagram**

The MS(n)A bit defines if the Dual Edge Capture mode is one-shot or continuous.

The ELS(n)B:ELS(n)A bits select the edge that is captured by channel (n), and ELS(n+1)B:ELS(n+1)A bits select the edge that is captured by channel (n+1). If both ELS(n)B:ELS(n)A and ELS(n+1)B:ELS(n+1)A bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the Dual Edge Capture mode, only channel (n) input is used and channel (n+1) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then CH(n)F bit is set and the channel (n) interrupt is generated (if CH(n)IE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input and (CH(n)F = 1), then CH(n+1)F bit is set and the channel (n+1) interrupt is generated (if CH(n+1)IE = 1).

The C(n)V register stores the value of FTM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of FTM counter when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, a coherency mechanism ensures coherent data when the C(n)V and C(n+1)V registers are read. The only requirement is that C(n)V must be read before C(n+1)V.

#### Note

- The CH(n)F, CH(n)IE, MS(n)A, ELS(n)B, and ELS(n)A bits are channel (n) bits.
- The CH(n+1)F, CH(n+1)IE, MS(n+1)A, ELS(n+1)B, and ELS(n+1)A bits are channel (n+1) bits.
- The Dual Edge Capture mode must be used with ELS(n)B:ELS(n)A = 0:1 or 1:0, ELS(n+1)B:ELS(n+1)A = 0:1 or 1:0 and the FTM counter in [Free running counter](#).

#### 23.5.24.1 One-Shot Capture mode

The One-Shot Capture mode is selected when (DECAPEN = 1), and (MS(n)A = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The ELS(n)B:ELS(n)A bits select the first edge to be captured, and ELS(n+1)B:ELS(n+1)A bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in One-Shot Capture mode, first the CH(n)F and CH(n+1) bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

### **23.5.24.2 Continuous Capture mode**

The Continuous Capture mode is selected when (DECAPEN = 1), and (MS(n)A = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The ELS(n)B:ELS(n)A bits select the initial edge to be captured, and ELS(n+1)B:ELS(n+1)A bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the CH(n)F and CH(n+1)F bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the CH(n+1)F bit. Therefore, when the CH(n+1)F bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

For a new sequence of the measurements in the Dual Edge Capture – Continuous mode, clear the CH(n)F and CH(n+1)F bits to start new measurements.

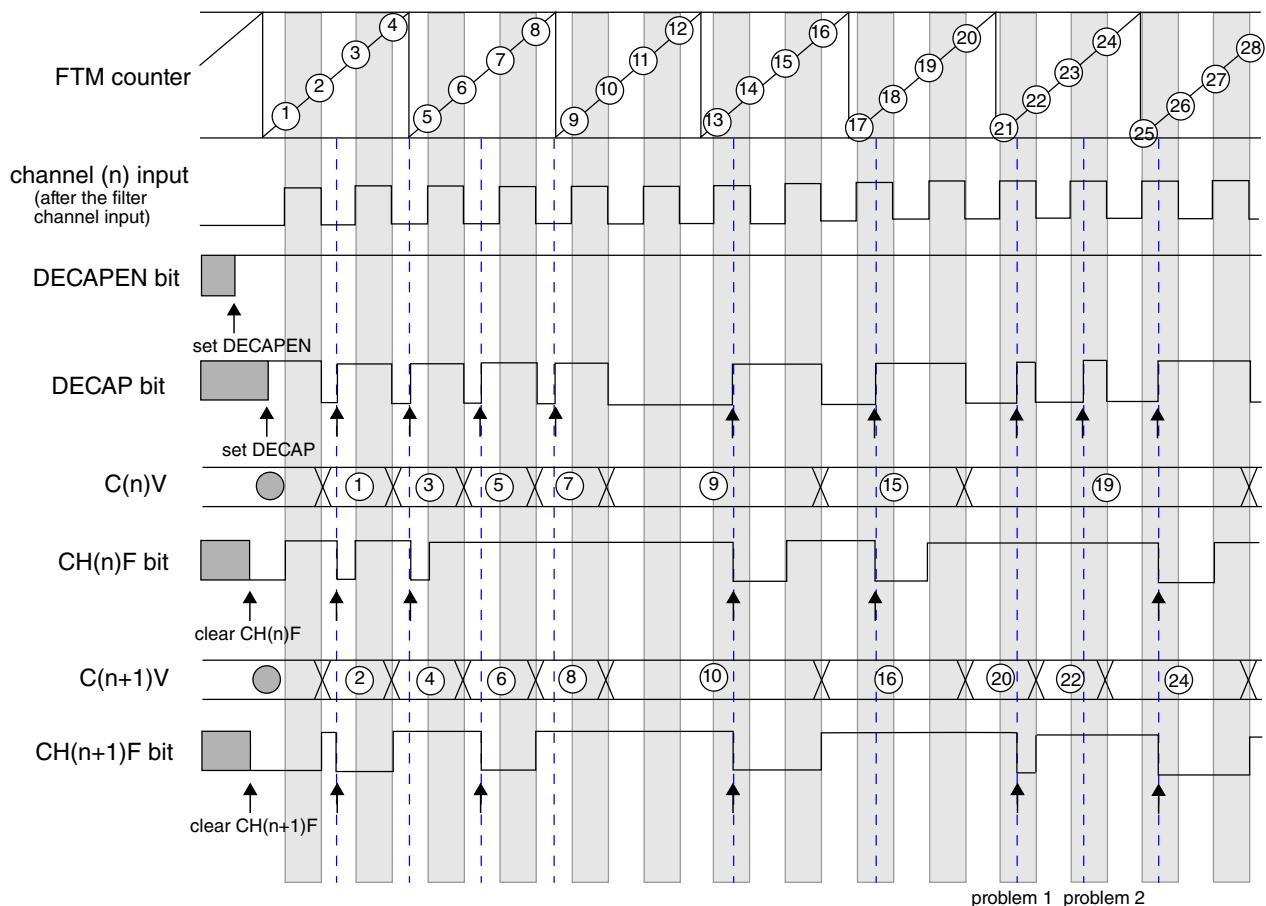
### **23.5.24.3 Pulse width measurement**

If the channel (n) is configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1) and the channel (n+1) to capture falling edges (ELS(n+1)B:ELS(n+1)A = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0) and the channel (n+1) to capture rising edges (ELS(n+1)B:ELS(n+1)A = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in [One-Shot Capture mode](#) or [Continuous Capture mode](#).

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next

positive polarity pulse width. The CH(n)F bit is set when the first edge of this pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second edge of this pulse is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



#### Note

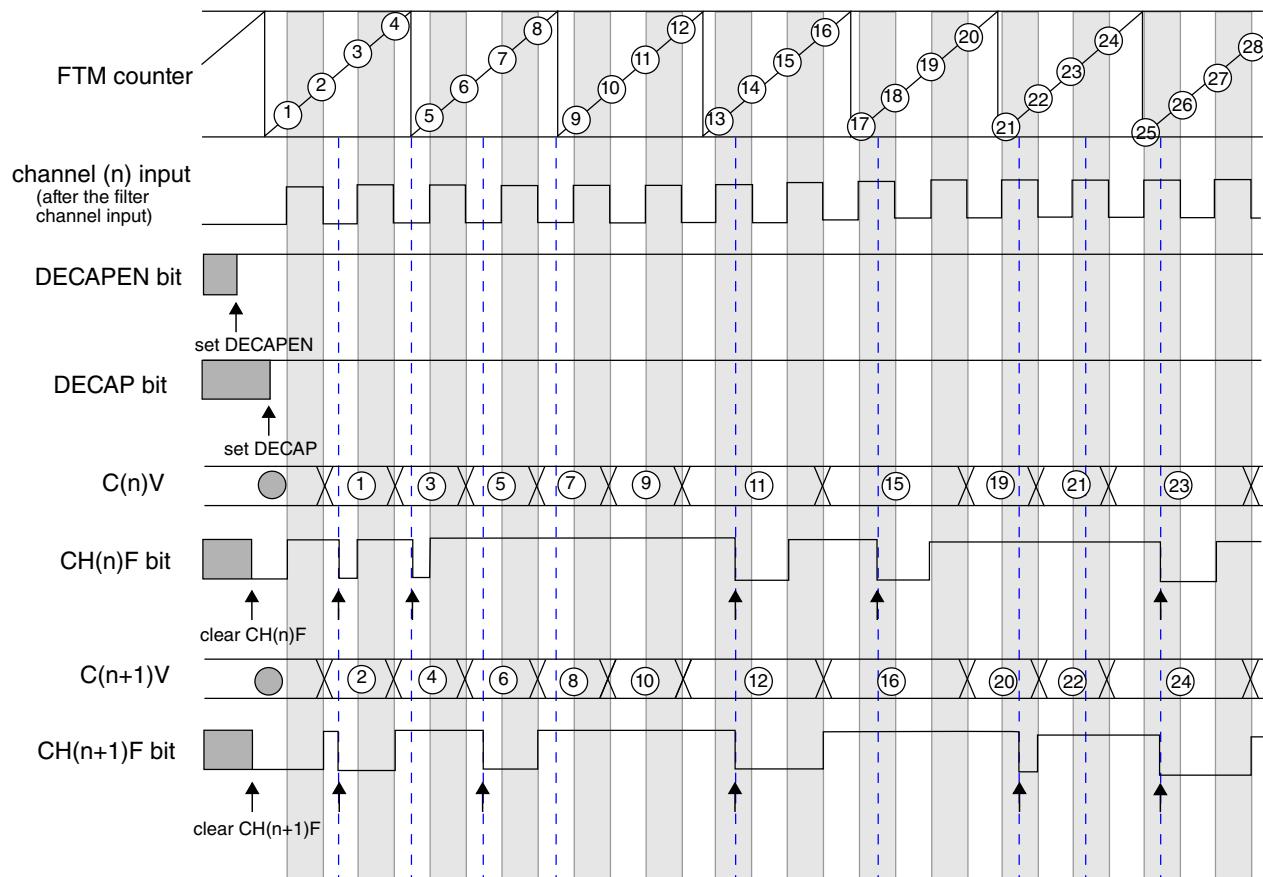
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

**Figure 23-82. Dual Edge Capture – One-Shot mode for positive polarity pulse width measurement**

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit

## Functional description

is set when the second edge of this pulse is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. The CH(n+1)F bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



### Note

- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

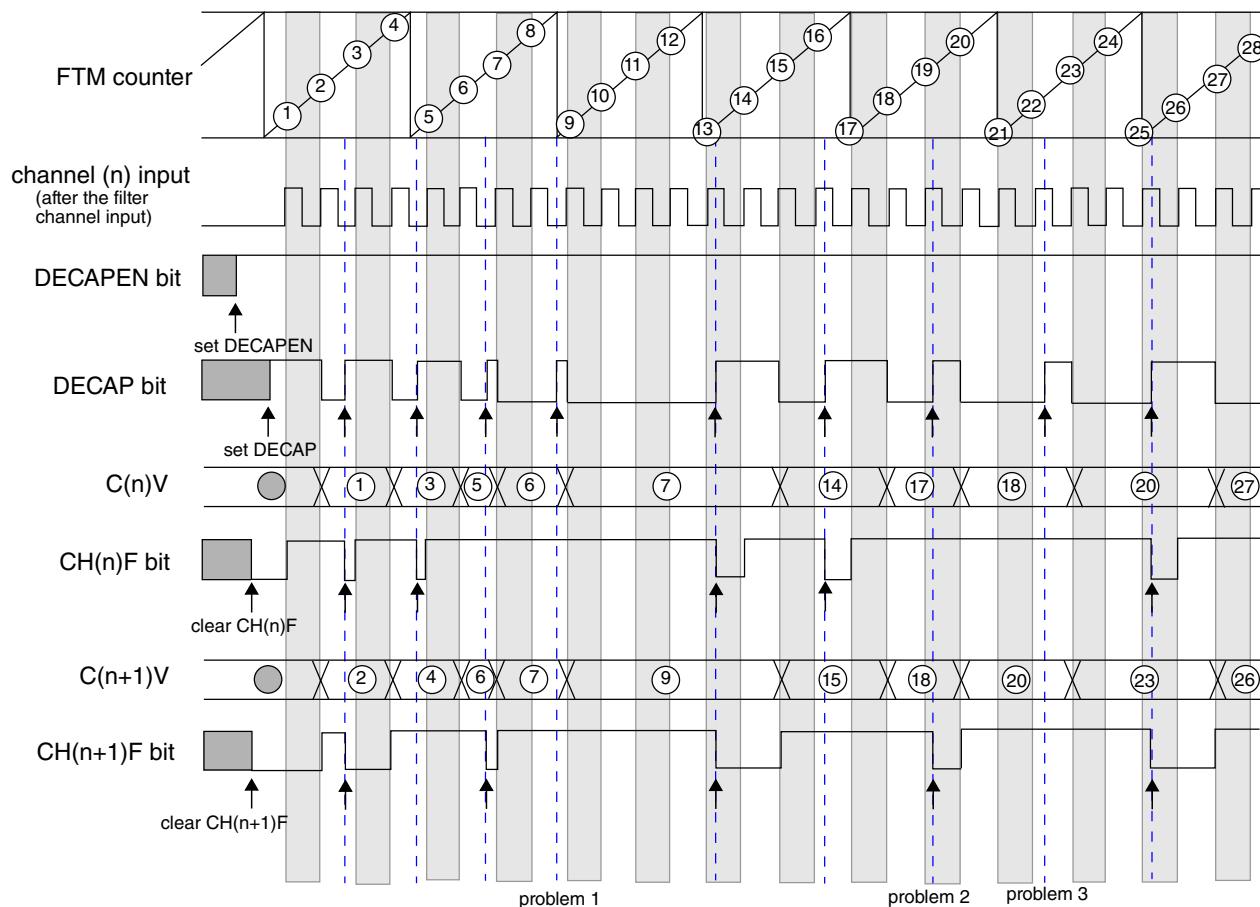
**Figure 23-83. Dual Edge Capture – Continuous mode for positive polarity pulse width measurement**

### 23.5.24.4 Period measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1 and ELS(n+1)B:ELS(n+1)A = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0 and ELS(n+1)B:ELS(n+1)A = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in [One-Shot Capture mode](#) or [Continuous Capture mode](#).

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next period. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.



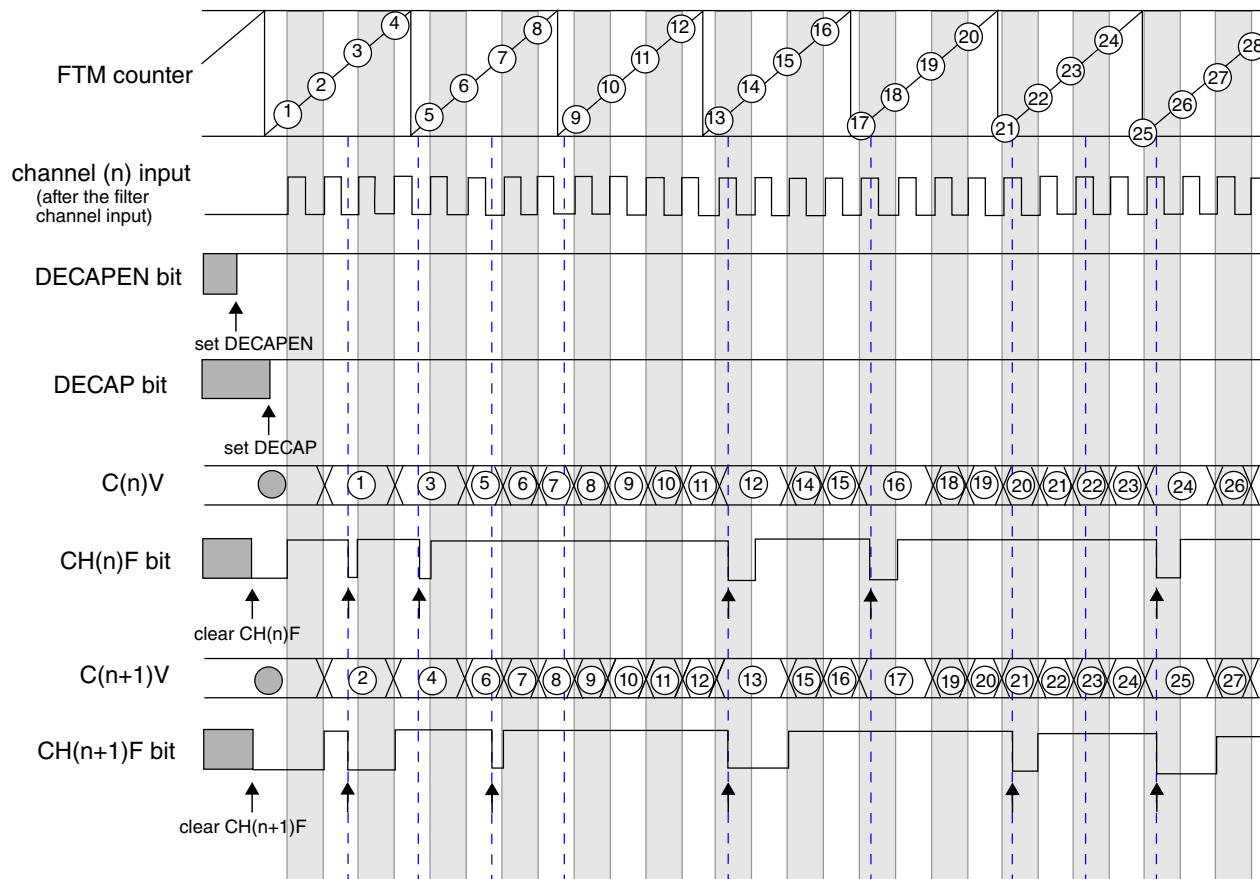
#### Note

- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 3: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

**Figure 23-84. Dual Edge Capture – One-Shot mode to measure of the period between two consecutive rising edges**

## Functional description

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set when the second rising edge is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. The CH(n+1)F bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.



### Note

- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

**Figure 23-85. Dual Edge Capture – Continuous mode to measure of the period between two consecutive rising edges**

### 23.5.24.5 Read coherency mechanism

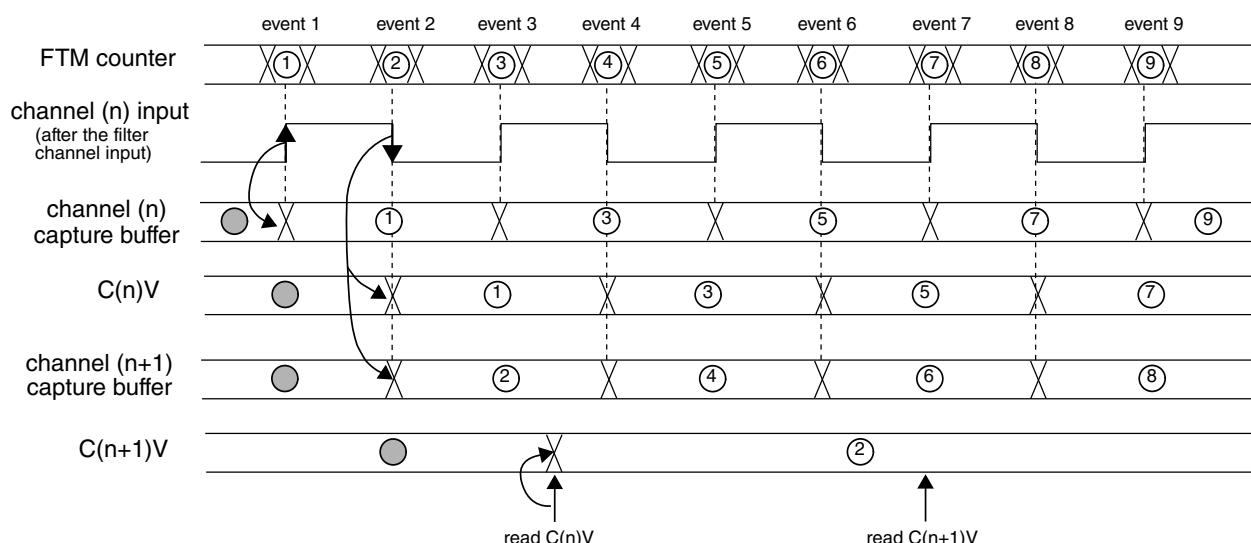
The Dual Edge Capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n

+1) are in Dual Edge Capture – Continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal. C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

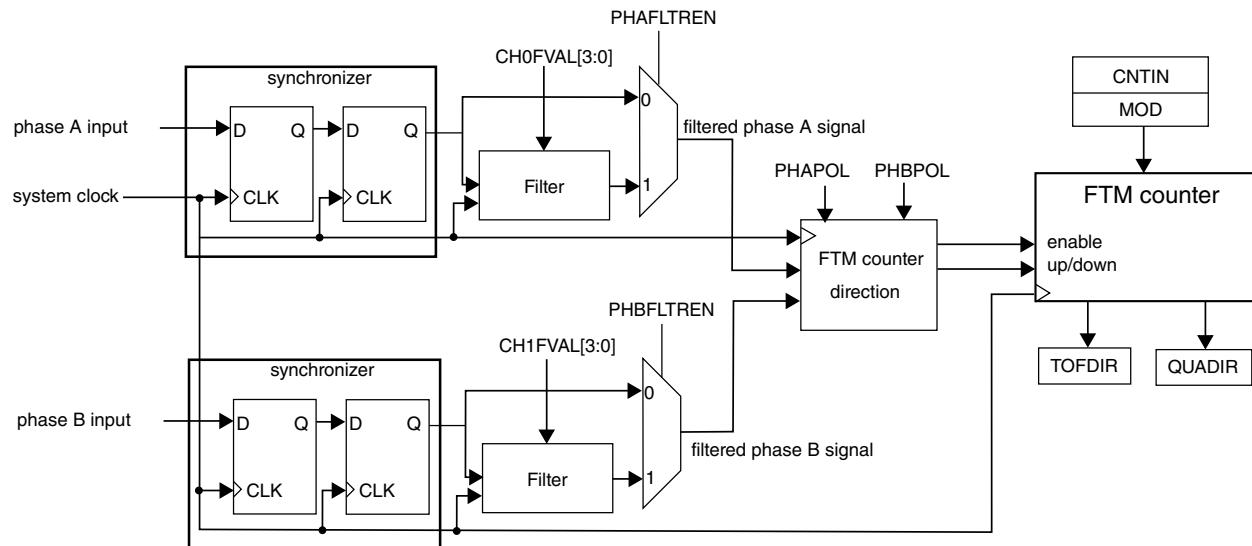


**Figure 23-86. Dual Edge Capture mode read coherency mechanism**

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

### 23.5.25 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement. The following figure shows the quadrature decoder block diagram.



**Figure 23-87. Quadrature Decoder block diagram**

Each one of input signals phase A and B has a filter that is equivalent to the filter used in the channels input; [Filter for Input Capture mode](#). The phase A input filter is enabled by PHAFLTREN bit and this filter's value is defined by CH0FVAL[3:0] bits (CH(n)FVAL[3:0] bits in FILTER0 register). The phase B input filter is enabled by PHBFLTREN bit and this filter's value is defined by CH1FVAL[3:0] bits (CH(n+1)FVAL[3:0] bits in FILTER0 register).

Except for CH0FVAL[3:0] and CH1FVAL[3:0] bits, no channel logic is used in Quadrature Decoder mode.

### Note

Notice that the FTM counter is clocked by the phase A and B input signals when quadrature decoder mode is selected.

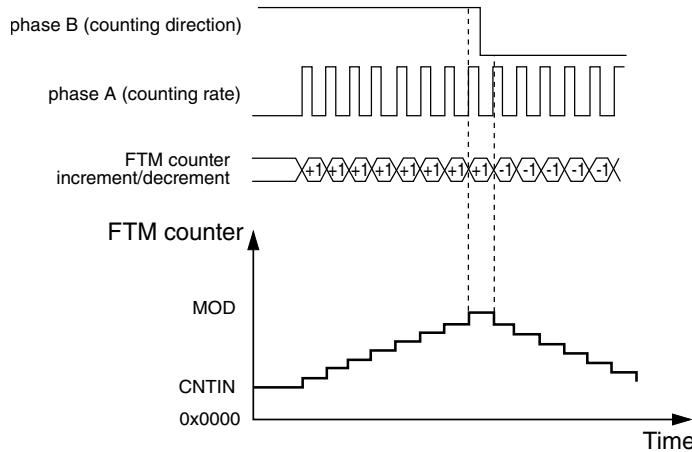
Therefore it is expected that the Quadrature Decoder be used only with the FTM channels in input capture or output compare modes.

### Note

An edge at phase A must not occur together an edge at phase B and vice-versa.

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the phase B input value indicates the counting direction, and the phase A input defines the counting rate. The FTM counter is updated when there is a rising edge at phase A input signal.



**Figure 23-88. Quadrature Decoder – Count and Direction Encoding mode**

If QUADMODE = 0, then the Phase A and Phase B Encoding mode is enabled; see the following figure. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate. The FTM counter is updated when there is an edge either at the phase A or phase B signals.

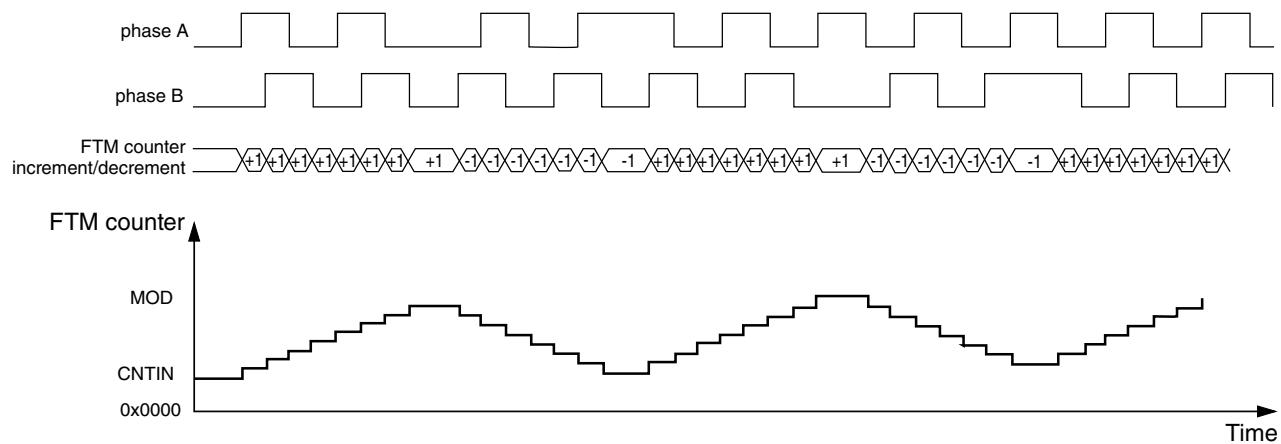
If PHAPOL = 0 and PHBPOL = 0, then the FTM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one;

and the FTM counter decrement happens when:

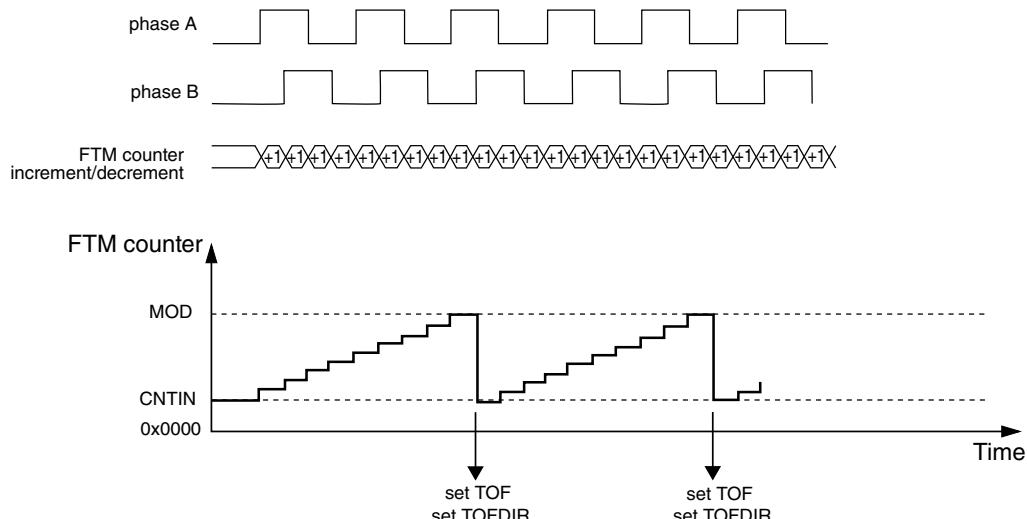
- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

## Functional description



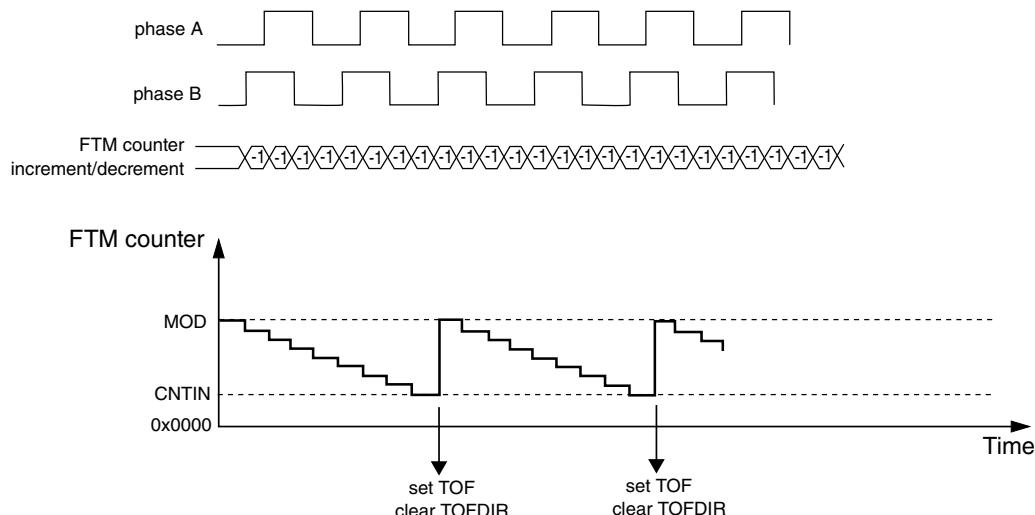
**Figure 23-89. Quadrature Decoder – Phase A and Phase B Encoding mode**

The following figure shows the FTM counter overflow in up counting. In this case, when the FTM counter changes from MOD to CNTIN, TOF and TOFDIR bits are set. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was up when the FTM counter overflow occurred.



**Figure 23-90. FTM Counter overflow in up counting for Quadrature Decoder mode**

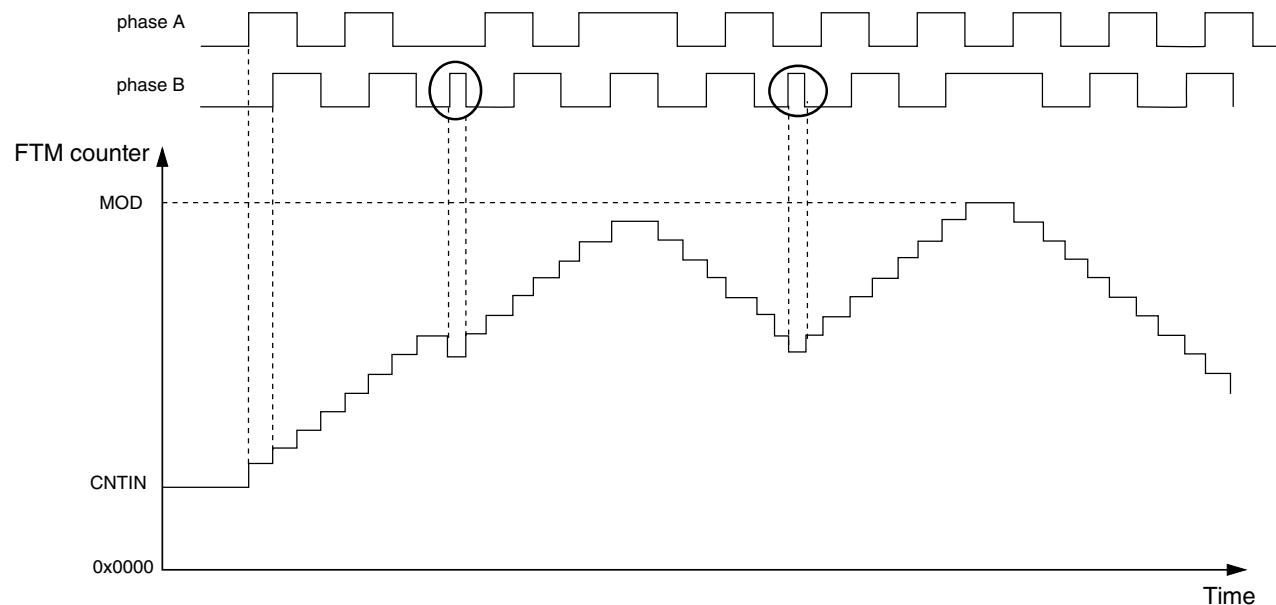
The following figure shows the FTM counter overflow in down counting. In this case, when the FTM counter changes from CNTIN to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was down when the FTM counter overflow occurred.



**Figure 23-91. FTM counter overflow in down counting for Quadrature Decoder mode**

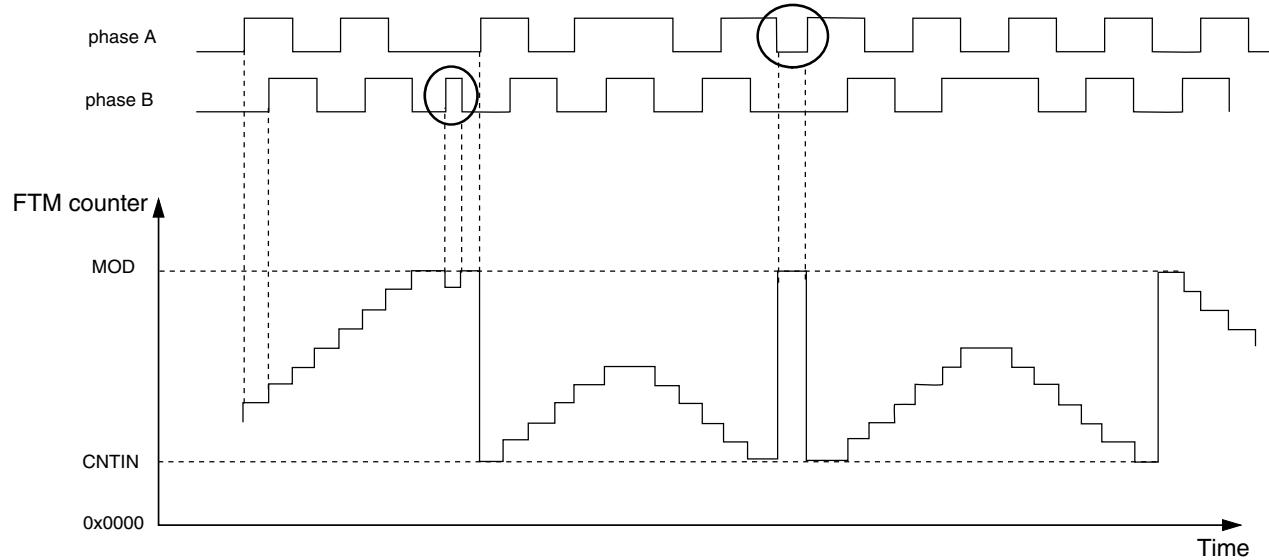
### 23.5.25.1 Quadrature Decoder boundary conditions

The following figures show the FTM counter responding to motor jittering typical in motor position control applications.



**Figure 23-92. Motor position jittering in a mid count value**

The following figure shows motor jittering produced by the phase B and A pulses respectively:



**Figure 23-93. Motor position jittering near maximum and minimum count value**

The first highlighted transition causes a jitter on the FTM counter value near the maximum count value (MOD). The second indicated transition occurs on phase A and causes the FTM counter transition between the maximum and minimum count values which are defined by MOD and CNTIN registers.

The appropriate settings of the phase A and phase B input filters are important to avoid glitches that may cause oscillation on the FTM counter value. The preceding figures show examples of oscillations that can be caused by poor input filter setup. Thus, it is important to guarantee a minimum pulse width to avoid these oscillations.

### 23.5.26 Intermediate load

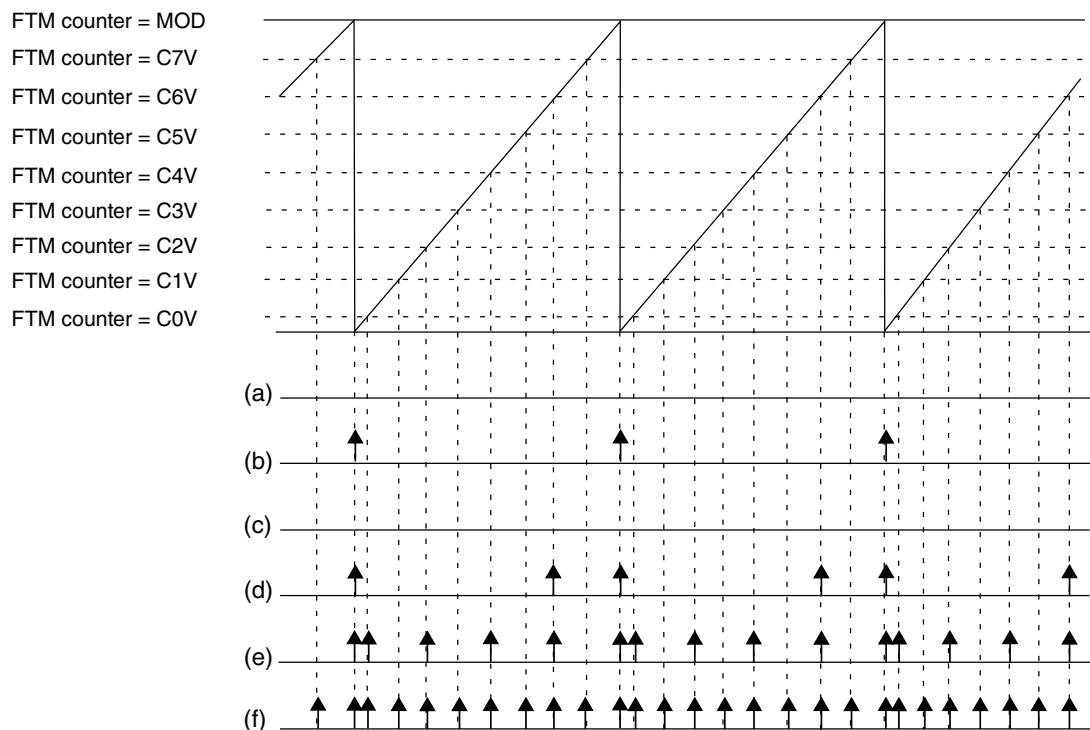
The PWMLOAD register allows to update the MOD, CNTIN, and C(n)V registers with the content of the register buffer at a defined load point. In this case, it is not required to use the PWM synchronization.

There are multiple possible loading points for intermediate load:

**Table 23-19. When possible loading points are enabled**

Loading point	Enabled
When the FTM counter wraps from MOD value to CNTIN value	Always
At the channel (j) match (FTM counter = C(j)V)	When CHjSEL = 1

The following figure shows some examples of enabled loading points.

**NOTE**

- (a) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
- (b) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
- (c) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 1, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
- (d) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
- (e) LDOK = 1, CH0SEL = 1, CH1SEL = 0, CH2SEL = 1, CH3SEL = 0, CH4SEL = 1, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
- (f) LDOK = 1, CH0SEL = 1, CH1SEL = 1, CH2SEL = 1, CH3SEL = 1, CH4SEL = 1, CH5SEL = 1, CH6SEL = 1, CH7SEL = 1

**Figure 23-94. Loading points for intermediate load**

After enabling the loading points, the LDOK bit must be set for the load to occur. In this case, the load occurs at the next enabled loading point according to the following conditions:

**Table 23-20. Conditions for loads occurring at the next enabled loading point**

When a new value was written	Then
To the MOD register	The MOD register is updated with its write buffer value.
To the CNTIN register and CNTINC = 1	The CNTIN register is updated with its write buffer value.
To the C(n)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n)V register is updated with its write buffer value.
To the C(n+1)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1)	The C(n+1)V register is updated with its write buffer value.

**NOTE**

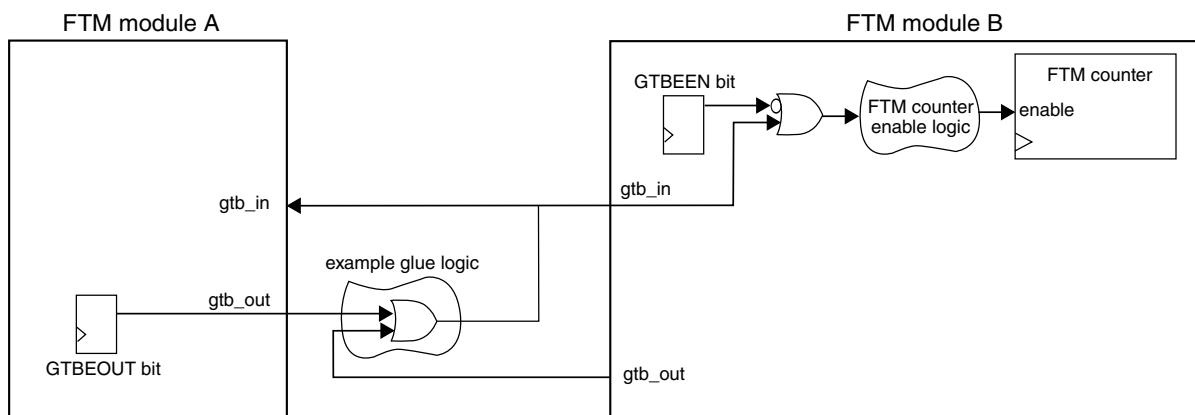
- If ELSjB and ELSjA bits are different from zero, then the channel (j) output signal is generated according to the configured output mode. If ELSjB and ELSjA bits are zero,

then the generated signal is not available on channel (j) output.

- If  $\text{CHjIE} = 1$ , then the channel (j) interrupt is generated when the channel (j) match occurs.
- At the intermediate load neither the channels outputs nor the FTM counter are changed. Software must set the intermediate load at a safe point in time.

### 23.5.27 Global time base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.



**Figure 23-95. Global time base (GTB) block diagram**

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the input signal  $gtb\_in$ , and the output signal  $gtb\_out$ . The GTBEEN bit enables  $gtb\_in$  to control the FTM counter enable signal:

- If  $\text{GTBEEN} = 0$ , each one of FTM modules works independently according to their configured mode.
- If  $\text{GTBEEN} = 1$ , the FTM counter update is enabled only when  $gtb\_in$  is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the  $gtb\_out$  signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the  $gtb\_in$  and  $gtb\_out$  signals, represented by the example glue logic shown in the figure. Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the chip-specific FTM information for the chip's specific implementation.

**NOTE**

- In order to use the GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb\_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation. The GTB feature only allows the FTM counters to *start* their operation synchronously.

**23.5.27.1 Enabling the global time base (GTB)**

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM to the intended configuration. The FTM counter mode needs to be consistent across all participating modules.
3. Write 1 to CONF[GTBEEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. The clock source needs to be consistent across all participating modules.
5. Reset the FTM counter: Write any value to the CNT register.

To initiate the GTB feature in the configuration described in the preceding figure, write 1 to CONF[GTBEOUT] in the FTM module used as the time base.

**23.6 Reset overview**

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 00b);
- the timer overflow interrupt is zero, see [Timer Overflow Interrupt](#);
- the channels interrupts are zero, see [Channel \(n\) Interrupt](#);
- the fault interrupt is zero, see [Fault Interrupt](#);
- the channels are in input capture mode, see [Input Capture mode](#);
- the channels outputs are zero;
- the channels pins are not controlled by FTM (ELS(n)B:ELS(n)A = 0:0) (See the table in the description of CnSC register).

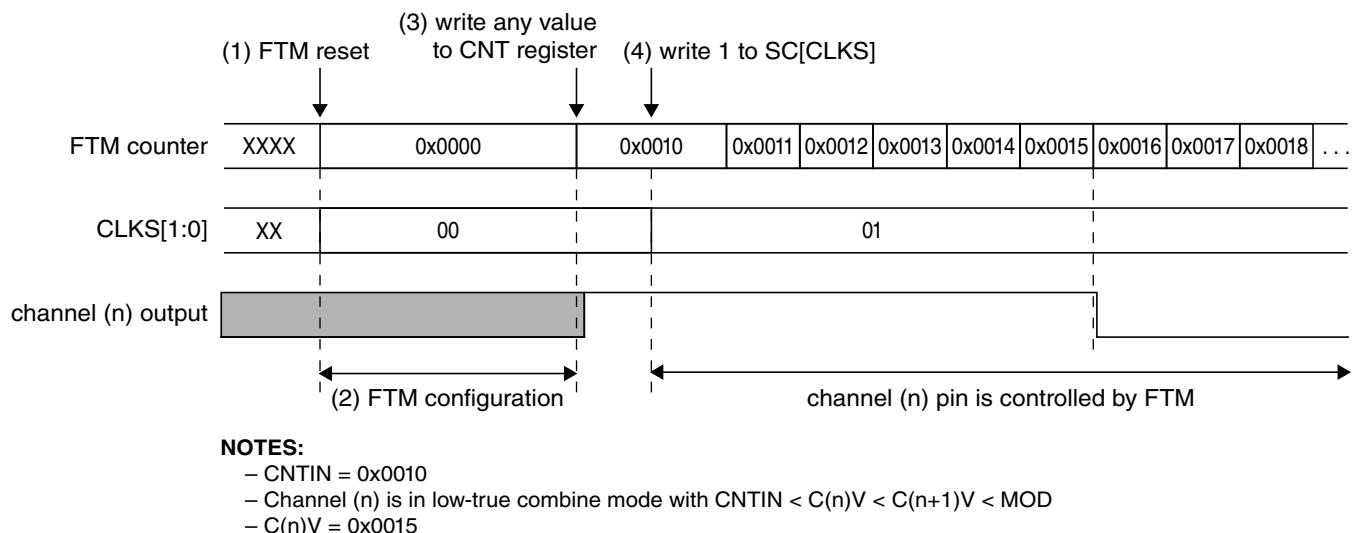
## Reset overview

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (see the description of the CLKS field in the Status and Control register), its value is updated to zero and the pins are not controlled by FTM (See the table in the description of CnSC register).

After the reset, the FTM should be configured (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

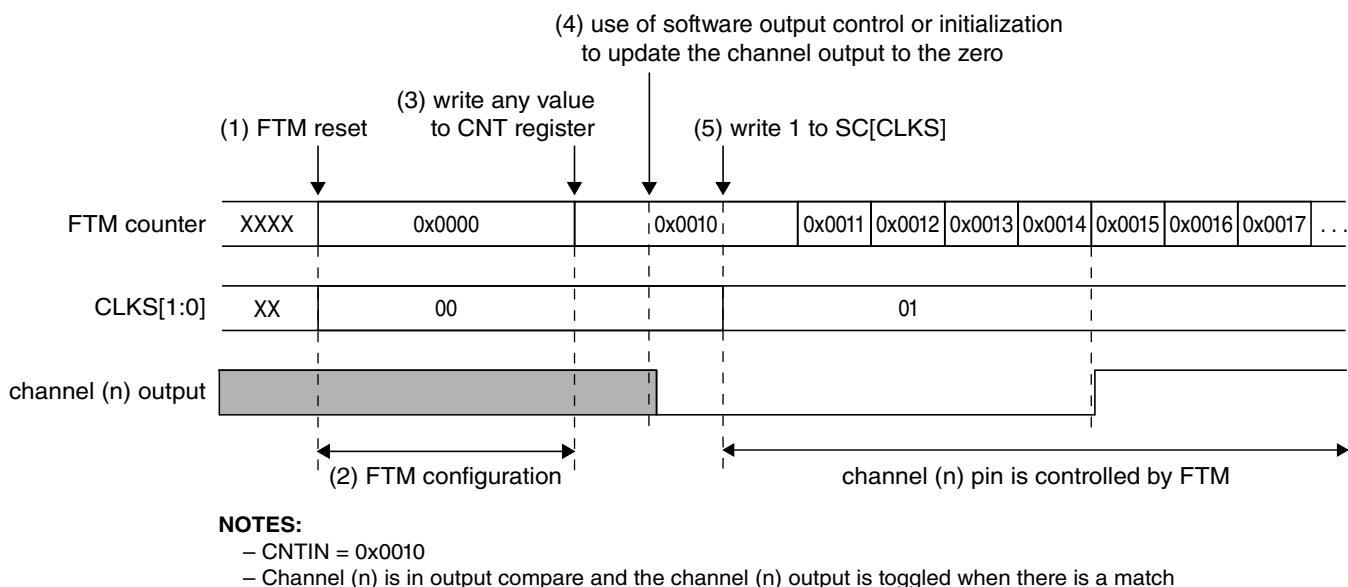
Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) ([Counter reset](#)).

The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero (See the table in the description of CnSC register).



**Figure 23-96. FTM behavior after reset when the channel (n) is in Combine mode**

The following figure shows an example when the channel (n) is in Output Compare mode and the channel (n) output is toggled when there is a match. In the Output Compare mode, the channel output is not updated to its initial value when there is a write to CNT register (item 3). In this case, use the software output control ([Software output control](#)) or the initialization ([Initialization](#)) to update the channel output to the selected value (item 4).



**Figure 23-97. FTM behavior after reset when the channel (n) is in Output Compare mode**

## 23.7 FTM Interrupts

### 23.7.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

### 23.7.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

### 23.7.3 Fault Interrupt

The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

## 23.8 Initialization Procedure

The following initialization procedure is recommended to configure the FlexTimer operation. This procedure can also be used to do a new configuration of the FlexTimer operation.

- Define the POL bits.
- Mask the channels outputs using SYNCHOM = 0. Two clocks after the write to OUTMASK, the channels output are in the safe value.
- (Re)Configuration FTM counter and channels to generation of periodic signals - Disable the clock. If the selected mode is Quadrature Decoder, then disable this mode. Examples of the (re)configuration:
  - Write to MOD.
  - Write to CNTIN.
  - Select OC, EPWM, CPWM, Combine, Complement modes for all channels that will be used
  - Select the high-true and low-true channels modes.
  - Write to CnV for all channels that will be used .
  - (Re)Configure deadtime and fault control.
  - Do not use the SWOC without SW synchronization (see item 6).
  - Do not use the Inverting without SW synchronization (see item 6).
  - Do not use the Initialization.
  - Do not change the polarity control.
  - Do not configure the HW synchronization
- Write any value to CNT. The FTM Counter is reset and the channels output are updated according to new configuration.
- Enable the clock. Write to CLKS[1:0] bits a value different from zero. If in the Quadrature Decoder mode, enable this mode.
- Configure the SW synchronization for SWOC (if it is necessary), Inverting (if it is necessary) and Output Mask (always)
  - Select synchronization for Output Mask Write to SYNC (SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
  - Write to SYNCONF.
    - HW Synchronization can not be enabled (HWSOC = 0, HWINV = 0, HWOM = 0, HWWRBUF = 0, HWRSTCNT = 0, HWTRIGMODE = 0).
    - SW Synchronization for SWOC (if it is necessary): SWSOC = [0/1] and SWOC = [0/1].
    - SW Synchronization for Inverting (if it is necessary): SWINV = [0/1] and INV = [0/1].
    - SW Synchronization for SWOM (always): SWOM = 1. No enable the SW Synchronization for write buffers (because the writes to registers with write buffer are done using CLKS[1:0] = 2'b00): SWWRBUF = 0 and CNTINC = 0.
    - SW Synchronization for counter reset (always): SWRSTCNT = 1.
    - Enhanced synchronization (always): SYNCMODE = 1

- If the SWOC is used ( $SWSOC = 1$  and  $SWOC = 1$ ), then write to SWOCTRL register.
- If the Inverting is used ( $SWINVC = 1$  and  $INVC = 1$ ), then write to INVCTRL register.
- Write to OUTMASK to enable the masked channels.
- Generate the Software Trigger Write to SYNC ( $SWSYNC = 1$ ,  $TRIG2 = 0$ ,  $TRIG1 = 0$ ,  $TRIGO = 0$ ,  $SYNCHOM = 1$ ,  $REINIT = 0$ ,  $CNTMAX = 0$ ,  $CNTMIN = 0$ )



# Chapter 24

## General Purpose I/O (GPIO)

### 24.1 The GPIO module as implemented on the chip

This section provides details about how the GPIO module is integrated into this chip.

**Table 24-1. GPIO module as implemented on chip**

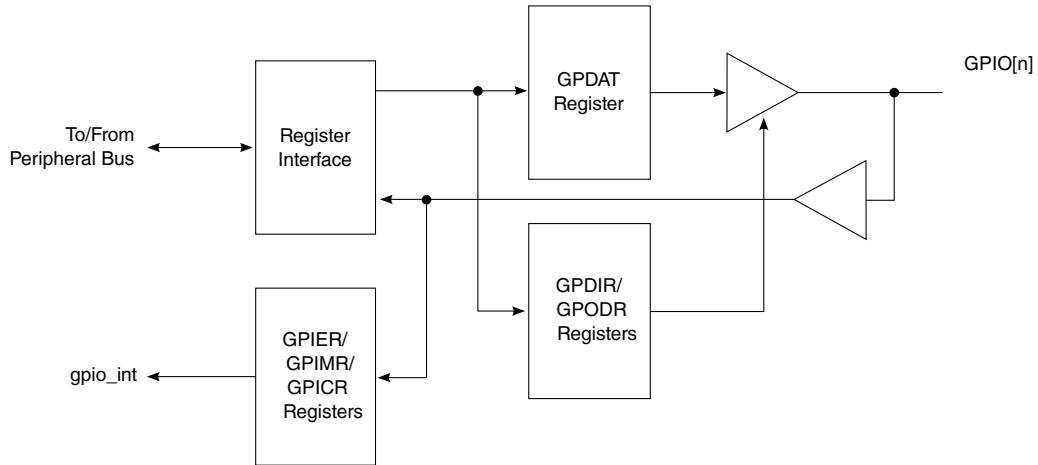
Module name	Module base address	Supported <sup>1</sup> ports	Number of ports
GPIO1	0x230_0000	13:31 (GPIO1[13] is output only signal)	19
GPIO2	0x231_0000	0:15, 25:27	19
GPIO3	0x232_0000	0:27	28
GPIO4	0x233_0000	0:3,10:13, 29:30	10

1. GPIO signals are typically multiplexed with other signals. “Supported” in this context means that any signal multiplexing configuration has selected GPIO functionality. See the Signals chapter for signal multiplexing details.

### 24.2 GPIO overview

This chapter describes the general-purpose I/O (GPIO) module, including signal descriptions, register settings and interrupt capabilities.

This figure shows the block diagram for the GPIO module.

**Figure 24-1. GPIO module block diagram**

In general, the GPIO module supports up to 32 general-purpose I/O ports. Each port can be configured as an input or as an output. However, some implementations may restrict specific ports to input-only, output-only, or reserved (unimplemented). See "The GPIO module as implemented on the chip" section for more information. If a port is configured as an input, it can optionally generate an interrupt upon detection of a change. If a port is configured as an output, it can be individually configured as an open-drain or a fully active output.

## 24.3 GPIO features summary

The GPIO unit includes the following features:

- Supports 32 general-purpose input/output ports
- All signals are high-impedance during reset
- Open-drain capability on all ports
- All ports can optionally generate an interrupt upon changing their state
- Ports may be multiplexed with other functional signals

## 24.4 GPIO signal descriptions

This table provides detailed descriptions of the external GPIO signals. Note that depending on the signal multiplexing, some signals may not be available.

**Table 24-2. GPIO external signals-detailed signal descriptions**

Signal	I/O	Description
GPIO[0:31]	I/O	<p>General Purpose I/O. Each signal can be individually set to act as input or output, according to application needs.</p> <p>Some implementations may restrict specific ports to input-only, output-only, or reserved (unimplemented). See "The GPIO module as implemented on the chip" section for more information.</p>
	<b>State Meaning</b>	Asserted/Negated-Defined per application.
	<b>Timing</b>	<p>Assertion/Negation-Inputs can be asserted completely asynchronously.</p> <p>Outputs are asynchronous to any externally visible clock.</p>

## 24.5 GPIO register descriptions

The programmable register map for the GPIO module occupies 28 bytes of memory-mapped space. The full register address is comprised of the base address (specified in CCSR address space) plus the module base address, plus the specific register's offset within the module. The table below shows the memory map for the GPIO module.

All GPIO registers are 32 bits wide located on 32-bit address boundaries. Note that reading undefined portions of the memory map returns all zeros and writing has no effect.

### 24.5.1 GPIO memory map

GPIO1 base address: 230\_0000h

GPIO2 base address: 231\_0000h

GPIO3 base address: 232\_0000h

GPIO4 base address: 233\_0000h

## GPIO register descriptions

Offset	Register	Width (In bits)	Access	Reset value
0h	GPIO direction register (GPDIR)	32	RW	0000_0000h
4h	GPIO open drain register (GPODR)	32	RW	0000_0000h
8h	GPIO data register (GPDAT)	32	RW	0000_0000h
Ch	GPIO interrupt event register (GPIER)	32	W1C	<a href="#">Table 24-2</a>
10h	GPIO interrupt mask register (GPIMR)	32	RW	0000_0000h
14h	GPIO interrupt control register (GPICR)	32	RW	0000_0000h

## 24.5.2 GPIO direction register (GPDIR)

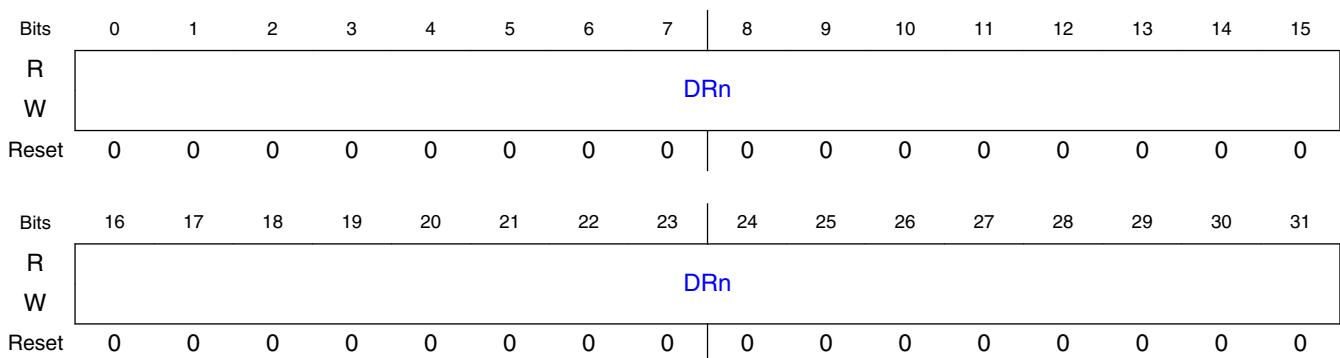
### 24.5.2.1 Offset

Register	Offset
GPDIR	0h

### 24.5.2.2 Function

The GPIO direction register (GPDIR) defines the direction of the individual ports.

### 24.5.2.3 Diagram



## 24.5.2.4 Fields

Field	Function
0-31	Direction. Indicates whether a pin is used as an input or an output.
DRn	00000000000000000000000000000000b - The corresponding pin is an input. 00000000000000000000000000000001b - The corresponding pin is an output.

## 24.5.3 GPIO open drain register (GPODR)

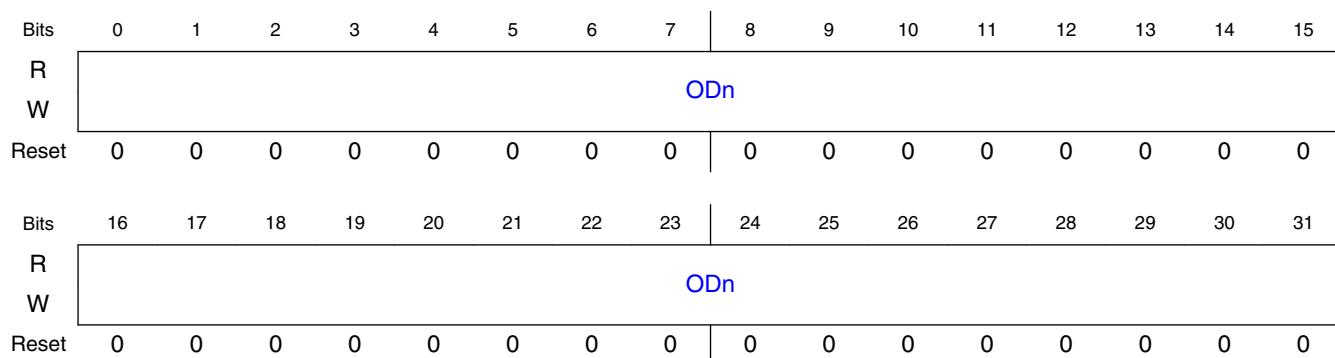
### 24.5.3.1 Offset

Register	Offset
GPODR	4h

### 24.5.3.2 Function

The GPIO open drain register (GPODR) defines the way individual ports drive their output.

### 24.5.3.3 Diagram



## 24.5.3.4 Fields

Field	Function
0-31 ODn	Open drain configuration. Indicates whether a signal is actively driven as an output or is an open-drain driver. This register has no effect on signals programmed as inputs in GPDIR.  00000000000000000000000000000000b - The corresponding signal is actively driven as an output. 00000000000000000000000000000001b - The corresponding signal is an open-drain driver. As an output, the signal is driven active-low, otherwise it is not driven (high impedance).

## 24.5.4 GPIO data register (GPDAT)

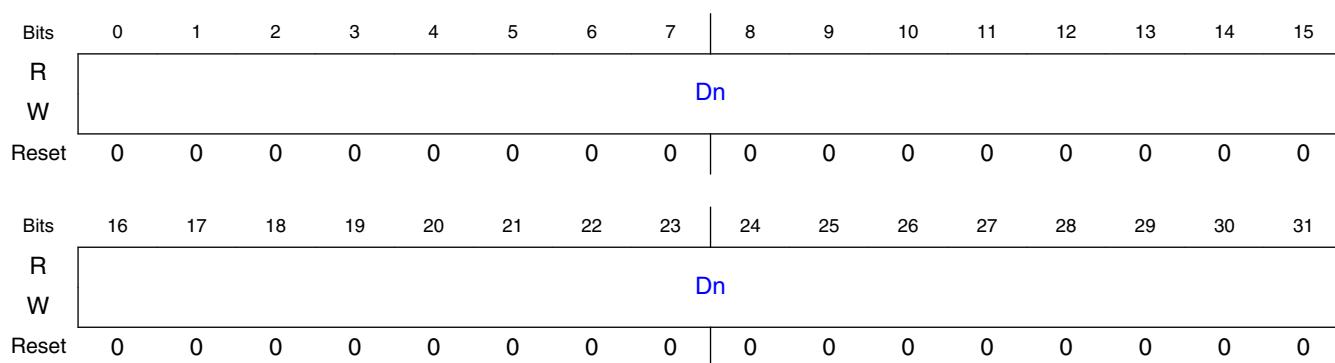
### 24.5.4.1 Offset

Register	Offset
GPDAT	8h

### 24.5.4.2 Function

The GPIO data register (GPDAT) carries the data in/out for the individual ports.

### 24.5.4.3 Diagram



#### 24.5.4.4 Fields

Field	Function
0-31 Dn	Data. Writes to this register latches the data which is presented on the external pins provided the corresponding GPDIR bit is configured as an output. When GPDIR is in output mode, GPDAT read operation returns data at pin. When GPDIR is in input mode, GPDAT read operation returns state of the port.

### 24.5.5 GPIO interrupt event register (GPIER)

#### 24.5.5.1 Offset

Register	Offset
GPIER	Ch

#### 24.5.5.2 Function

The GPIO interrupt event register (GPIER) carries information of the events that caused an interrupt. Each bit in GPIER, corresponds to an interrupt source. GPIER bits are cleared by writing ones. However, writing zero has no effect.

#### NOTE

Some implementations may ignore the interrupt mask as configured in GPIMR. In these implementations, a GPIER bit can be set even though the associated interrupt is masked. See The "GPIO module as implemented on the chip" section for more information.

### 24.5.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R									EVn								
W									W1C								
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R									EVn								
W									W1C								
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	

### 24.5.5.4 Fields

Field	Function
0-31	Interrupt events. Indicates whether an interrupt event occurred on the corresponding GPIO signal.
EVn	00000000000000000000000000000000b - No interrupt event occurred on the corresponding GPIO signal. 00000000000000000000000000000001b - An interrupt event occurred on the corresponding GPIO signal.

## 24.5.6 GPIO interrupt mask register (GPIMR)

### 24.5.6.1 Offset

Register	Offset
GPIMR	10h

### 24.5.6.2 Function

The GPIO interrupt mask register (GPIMR) defines the interrupt masking for the individual ports. When a masked interrupt request occurs, the corresponding GPIER bit is set, regardless of the GPIMR state. When one or more non-masked interrupt events occur, the GPIO module issues an interrupt to the interrupt controller.

### 24.5.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									IMn							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									IMn							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 24.5.6.4 Fields

Field	Function
0-31 IMn	Interrupt mask. Indicates whether an interrupt event is masked or not masked for the corresponding GPIO signal.  00000000000000000000000000000000b - The input interrupt signal is masked (disabled). 00000000000000000000000000000001b - The input interrupt signal is not masked (enabled).

## 24.5.7 GPIO interrupt control register (GPICR)

### 24.5.7.1 Offset

Register	Offset
GPICR	14h

### 24.5.7.2 Function

The GPIO interrupt control register (GPICR) determines whether the corresponding port line asserts an interrupt request upon either a high-to-low change or any change on the state of the signal.

### 24.5.7.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									EDn							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									EDn							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 24.5.7.4 Fields

Field	Function
0-31	Edge detection mode. The corresponding port line asserts an interrupt request according to the following:
EDn	00000000000000000000000000000000b - Any change on the state of the port generates an interrupt request. 00000000000000000000000000000001b - High-to-low change on the port generates an interrupt request.

# **Chapter 25**

## **Integrated Flash Controller (IFC)**

### **25.1 IFC overview**

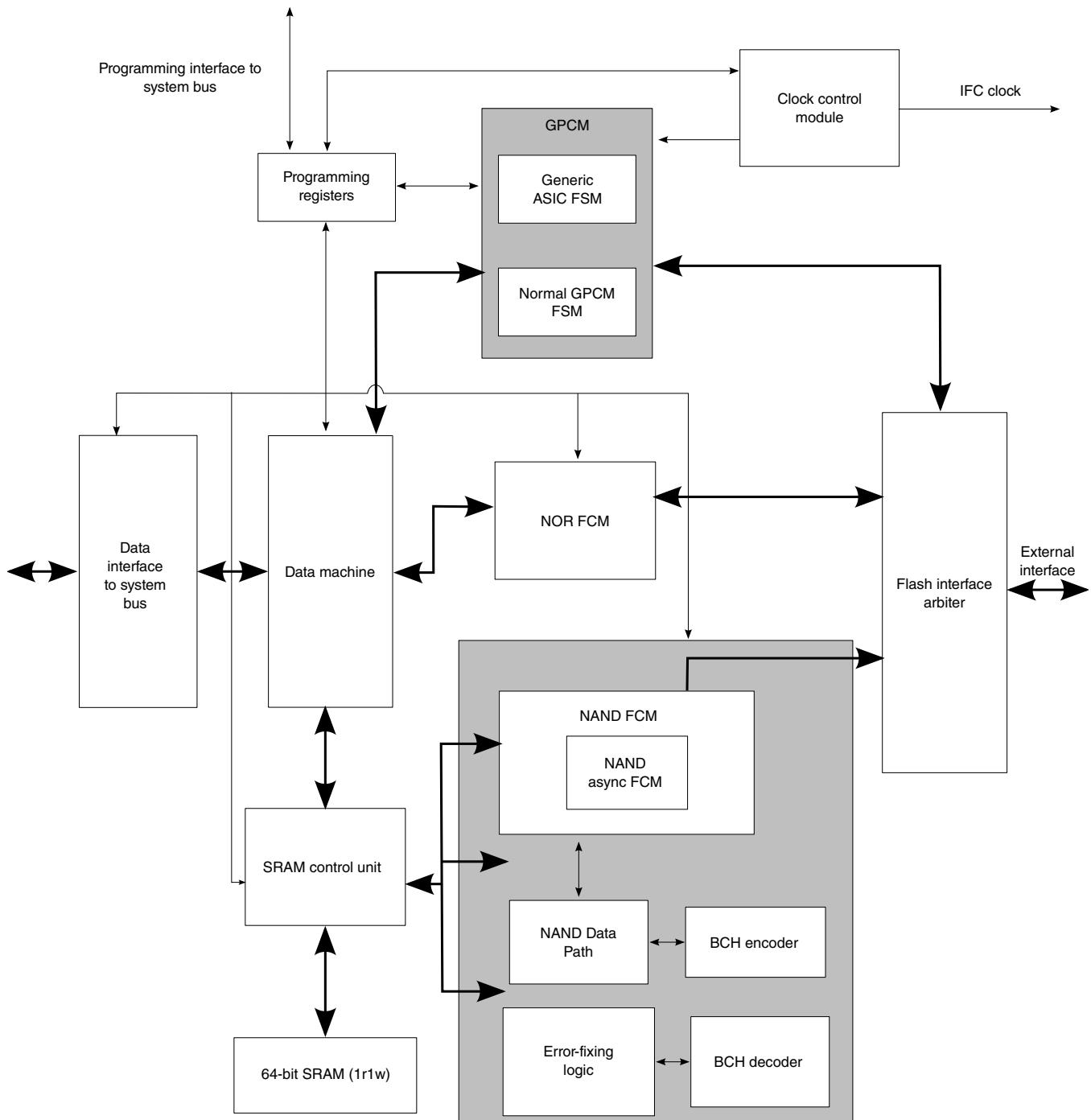
The integrated flash controller (IFC) is used to interface with external asynchronous/synchronous NAND flash, asynchronous NOR flash, SRAM, generic ASIC memory and EPROM.

It has seven chip-selects, to which a maximum of seven flash devices can be attached, although only one of these can be accessed at any given time.

The IFC handles pin multiplexing to the internal system bus based on the selected controller (NAND, NOR, GPCM, or generic ASIC). To save pins at the chip level, multiplexing of address pins can be done on the data bus using an address valid signal (AVD). The BCH error-correction algorithm is used to correct the error bits while reading from a NAND device.

This figure shows the block diagram of IFC.

## IFC overview



**Figure 25-1. IFC block diagram**

### 25.1.1 IFC features summary

The IFC supports both general and controller-specific features.

The general features of the IFC include the following:

- Flash controller with seven chip-selects
- Supports error and debug registers
- Functional muxing of pins between NAND, NOR, and GPCM
- Supports memory banks of sizes up to 256 MB (for NOR and GPCM)
- Write-protection capability (for NAND and NOR)
- Provision of software reset
- External transceiver enable/disable control on a per-bank basis

### 25.1.1.1 NAND flash controller features

The IFC NAND flash controller features are:

- x8/x16 NAND flash interface
- Support for ONFI-2.2 asynchronous interface (8-/16-bit)/NVDDR interface and mandatory commands
- BCH code for 4-bit and 8-bit error correction per sector of 512 bytes (using GF- $2^{13}$  Galois field) and 24 and 40-bit ECC per sector of 1 KB (using GF- $2^{14}$  Galois field):
  - For a 512-byte page and 16-byte spare region, 4-bit error correction is supported
  - For a 2 KB page and 64-byte spare region, 4-bit error correction is supported
  - For a 4 KB page width:
    - For 128-byte spare region, 4-bit error correction is supported
    - For 210-, 218-, and 224-byte spare regions, 4- or 8-bit error correction is supported
    - Spare size of minimum 176 bytes (8 for BBI + 4x42 for ECC bytes), 4-, 8-, or 24-bit BCH
    - Spare size of minimum 288 bytes (8 for BBI + 4x70 for ECC bytes), 4-, 8-, 24-, or 40-bit BCH
  - For an 8 KB page width:
    - At least 136(8 + 128) bytes of spare region is required for bad block information and ECC bytes, 4-bit BCH
    - 8 KB page, at least 264(8 + 256) bytes of spare region required for bad block information and ECC bytes, 4-bit or 8-bit BCH
    - Spare size of minimum 344 bytes (8 for BBI + 8x42 for ECC bytes), 4-, 8-, or 24-bit BCH
    - Spare size of minimum 568 bytes (8 for BBI + 8x70 for ECC bytes), 4-, 8-, 24-, or 40-bit BCH
  - For all ECC modes (4/8/24/40), parity bytes are stored at offset 08h in a spare region
- ECC generation/checking is optional

- Flexible timing control to allow interfacing with proprietary NAND devices
- Supports SLC and MLC flash devices with configurable page sizes of up to 8 KB
- Supports advance NAND commands such as cache, copy-back, and multi-plane programming
- Supports four RDY\_B/BSY\_B signals
- Programmable command and data transfer sequences of up to 15 steps
- Configurable block-size
- Interrupt for error handling and flash command completion event
- Internal SRAM of 9 KB
  - Memory-mapped
  - Can be configured as boot RAM
  - Acts as data buffer for normal operation
- Boot chip-select (CS0) available after system reset, with a boot block size of 8 KB for execute-in-place boot loading from NAND flash
- Supports flash devices of a magnitude of terabytes
  - By using 32 bit row address, 4 Giga pages of NAND flash can be accessed

### 25.1.1.2 NOR flash features

The IFC features a NOR flash controller.

- Compatible with asynchronous NOR flash (synchronous burst-read not supported)
- Provides memory-mapped interfacing to NOR
- Supports an address data multiplexed (ADM) NOR device
- Flexible timing control allows interfacing with proprietary NOR devices (WE\_B controlled writes only)
- Boot chip-select (CS0) is available at system reset
- Data bus width of 16-bits
- Burst size in NOR is governed by system bus size and length

### 25.1.1.3 GPCM and GASIC features

The IFC's general-purpose, chip-select Controller (GPCM) supports operation in either normal or generic ASIC modes.

Normal GPCM features include the following:

- Support for x8-/16-bit devices
- Compatible with general-purpose addressable device, such as SRAM and ROM
- External clock is generated with programmable division ratio (2, 3, 4, ... up to 16)
- Output enable signal (OE\_B)

- Write enable signal(s) (WE\_B)
- Even/odd parity data bus support
- External access termination signal (IFCTA\_B)
- Programmable burst support

Generic ASIC features include the following:

- Support for x8-/16-bit devices
- Address and data are shared on I/O bus
- The following address and data sequences are supported on the I/O bus:
  - 16-bit I/O: AADD
  - 8-bit I/O: AAAADDDD
- Configurable even/odd parity on address/data bus
- Parity-error detection

## 25.1.2 IFC modes of operation

The IFC contains one NAND controller, one NOR flash controller, and one GPCM/generic-ASIC controller.

It can be programmed such that all seven memory banks can work with NAND, NOR, and GPCM/generic-ASIC interfaces as required. However, only one memory bank can be active at any given time.

## 25.2 External signal descriptions

This section provides both general and detailed information on the chip's external signals.

This table provides an overview of the chip's external signals, and the following table then provides a much more detailed description of the signals.

**Table 25-1. Signal properties**

Chip signal name	IFC signal name	Alternate Functions	Mode	Description / Function	Number of signals	I/O
IFC_AD[0:15]	AD[0:15]	AD[0:15]	NAND FCM	I/O Bus for Asynchronous NAND	16	I/O
		AD[0:7]	NVDDR	I/O Bus for NVDDR sync NAND		
		AD[0:15]	NOR FCM	Bidirectional Data Bus for		

*Table continues on the next page...*

**Table 25-1. Signal properties (continued)**

Chip signal name	IFC signal name	Alternate Functions	Mode	Description / Function	Number of signals	I/O
				NOR. During AVD assertion, this bus will carry address bits. Address LSB or MSB will be governed by CSOR[ADM_SH FT_MODE]		
		AD[0:15]	GPCM	Bidirectional data bus for GPCM. For external address latching, this bus is used to carry address MSBs.		
		AD[0:15]	GASIC	Bidirectional shared address/data bus for GASIC.		
IFC_A[16:27]	ADDR[16:27]	-	-	Dedicated address bus used by NOR and GPCM mode	12	O
IFC_PAR[0:1]	PAR[0:1]	PAR[0:1]	GPCM	Parity data	2	I/O
		PAR[0:1]	GASIC	Parity address and data		
IFC_NDDDR_CLK	IFC_NDDDR_CLK	IFC_NDDDR_CLK	NV-DDR	External IFC DDR Clock (NVDDR mode)	1	O
		NDWE_B	Async NAND	NAND Write Enable	1	O
IFC_CS[0:6]_B	CE/CS[0:6]_B	-	-	Chip Selects	7	O
NDWE_B	Async	NAND Write Enable	-	Data buffer control	1	O
IFC_BCTL	BCTL	-				
IFC_TE	TE	-	-	External Transceiver Enable/Disable Control	1	I/O
IFC_NDDQS	NDDQS	-	NVDDR	Data Qualify Strobe to/from NAND memory (in NVDDR )	1	I/O
IFC_AVD	ALE/AVD	NDALE	NAND FCM	NAND Address Latch Enable	1	O

*Table continues on the next page...*

**Table 25-1. Signal properties (continued)**

Chip signal name	IFC signal name	Alternate Functions	Mode	Description / Function	Number of signals	I/O
		NRAVD	NOR FCM	NOR External Address Latch Enable/AVD		
		GPALE	GPCM	GPCM External Address Latch Enable		
IFC_CLE	CLE	NDCLE	NAND FCM	NAND Command Latch Enable	1	O
		GPWE1_B	GPCM	GPCM Write Byte Select 1		
IFC_OE_B	OE_B	NROE_B	NOR FCM	NOR Output Enable	1	O
		GPOE_B	GPCM	GPCM Output Enable		
		RW_L_B	GASIC	GASIC Read/Write Indication		
		NDRE_B	Async/NVDDR	NAND Read Enable		
IFC_WP[0:3]_B	WP[0:3]_B	NDWP[0:3]_B	NAND FCM	Per chip select based NAND Write Protect	4	O
IFC_RB[0:3]_B	RB[0:3]_B	NAND FCM	NAND Ready/Busy Input	4	I	NOR Ready Busy Input
		NRRB[0:3]_B	NOR FCM			
		IFCTA[0:3]_B	GPCM			GPCM External Termination
		RDY_L[0:3]_B	GASIC			
IFC_PERR_B	PERR_L_B	PERR_L_B	GASIC	Parity error from Generic ASIC	1	I
IFC_CLK[0:1]	CLK[0:1]	-	-	External IFC clock	2	O

This table describes the external signals in detail.

**Table 25-2. Detailed signal description**

Signal	I/O	Description	
AD[0:n]	I/O	Multiplexed address/data bus.	
		<b>State meaning</b>	Asserted/Negated - During the assertion of AVD, AD[0:n] are driven with the address for the access to follow. External logic should propagate the address on AD[0:n] while AVD is asserted and latch the address upon negation of AVD. After it is negated, AD[0:n] are either driven by write data or are switched to an input to sample read data driven by an external device. Following the last data transfer of a write access, AD[0:n] are released to a high-impedance state.  For exact functionality, behavior, and timings, refer to detailed waveform and timing diagrams within the specific interface.
ADDR[n:m]	O	Non-multiplexed address bus.	
		<b>State meaning</b>	Asserted/Negated - It is a non-multiplexed address bus which carries address MSBs or LSBs depending on CSOR[ADM_SHFT_MODE].
ALE	O	External Address Latch Enable (NAND) External Address Valid Signal (NOR) External Address Latch Enable(GPCM)  Control signal for an external address latch, which allows address and data to be multiplexed on the device pins	
		<b>State meaning</b>	Asserted - Latch enable is asserted with the address at the beginning of each memory controller transaction. The number of cycles for which it is asserted is governed by the FTIM0_CSn[EADC] field.  Negated - The exact timing of the negation of AVD is controlled by the FTIM0_CSn[EAHC] field. Note that no other control signals are asserted during the assertion of AVD.
CS[0:n]_B	O	Chip-Select; mutually exclusive chip-selects are available.	
		<b>State meaning</b>	Asserted/Negated - Used to enable each specific memory device connected to the IFC. CS_B[0] corresponds to the chip-select for memory bank 0, which has the memory type and attributes defined by CSPR0 and FTIM0_CS0.
WE[0]_B	O	NOR write enable, GPCM write byte select 0, GASIC start of frame (SOF_L_B).	
		<b>State meaning</b>	Asserted/Negated - Used to control the data write cycles on NOR flash.  For GPCM it validates its corresponding byte number on the data bus.  For GASIC it indicates start of a transaction frame.
NDQDS	I/O	Bi-directional Data Qualify Strobe for NAND in NV-DDR mode	
		<b>State Meaning</b>	Bi-directional signal used to capture write data in the NAND flash and read data in IFC. Data is captured/driven on both of its edges (rising and falling).  For program operations, the DQS is centred with respect to program data when driven by IFC.  For read operations, the DQS is edge-aligned with respect to read data when driven by NAND memory.
CLE	O	NAND Command latch enable, GPCM write byte select 1 which is WE[1]_B.	
		<b>State meaning</b>	Asserted/Negated - It enables the command cycle on NAND flash.  For GPCM it validates 1st byte(AD [8:15]) on data bus.
OE_B	O	NAND read enable, NOR output enable, GPCM output enable, GASIC Read/Write Enable and is valid only when SOF_L_B is asserted.	

*Table continues on the next page...*

**Table 25-2. Detailed signal description (continued)**

Signal	I/O	Description	
		<b>State meaning</b> Asserted/Negated - It enables data read cycles on NOR and GPCM devices. For GASIC, it indicates whether the transaction is read or write. High value on this indicates read operation while low value indicates write.	
WP[0:n]_B	O	Per chip select based non-muxed NAND Write Protect Output.	
		<b>State meaning</b> Asserted/Negated - It protects NAND flash from accidental erasure or program when asserted low. For GPCM it validates the corresponding byte number on data bus.	
RB[0:n]_B	I	NAND ready busy for all the chip-selects, NOR Read Busy, GPCM External Termination of Access, GASIC Ready Indication. Valid after the completion of address phase till it asserts.	
		<b>State meaning</b> Asserted/Negated - It indicates the ready busy status of the flash operation. It is used to stall the controller operation when the flash is indicating busy. For GPCM this input is used for termination of current access. For GASIC it indicates that the current GASIC transaction is accepted by device and the host can start the next transaction.	
BCTL	O	Data buffer control. Buffer control is disabled by FTIM0_CS0[BCTL0] field.	
		<b>State meaning</b> Asserted/Negated - The BCTL pin normally functions as a write (High)/read (Low) direction control for the external data buffer used in the AD lines (bidirectional bus). Default value of BCTL is high (write direction).	
TE	I/O	External transceiver enable/disable. This pin acts as an input during reset and should be weakly pulled up/down so as to disable the external transceiver during that time. After reset, the IFC drives TE with the value configured in CSPRn[TE] as per the selected chip select. When none of the CS pins are selected, TE is driven High-Z from the IFC (the external transceiver will be disabled by the appropriate external pull device).	
		<b>State meaning</b> Asserted/Negated - The TE pin functions as an enable/disable for an external transceiver connected to the AD lines. In asserted state, it enables the external transceiver and when negated, disables the transceiver.	
PAR[0:n]	I/O	GPCM data bus parity, GASIC address bus parity.	
		<b>State meaning</b> Asserted/Negated - For GPCM during Write a parity bit is generated for each data byte sent over the bus. Unused byte lanes have undefined parity. For GASIC a parity bit is sent for each address and data byte sent over the AD bus.	
IFC_NDDDR_CL K/NDWE_B	O	IFC NAND DDR External Clock for NV-DDR mode NAND Write Enable for ASync	
		<b>State Meaning</b> NV-DDR mode: External divided clock derived from IP clock for the NAND Flash. ASync Used to control write cycles for NAND Flash	
PERR_L_B	I	GASIC parity error indication. PERR_B is sampled a configurable number of IFC_CLK cycles after the assertion of SOF_L_B.	
		<b>State meaning</b> Asserted/Negated - Parity error is indicated by GASIC for wrong parity sent for address and data bytes.	
CLK[0:n]	O	IFC external clock.	
		<b>State meaning</b> This is an external clock derived from the IFC module input clock. Division ratio is programmable (using CCR register).	

## 25.2.1 Internal connectivity of WP/RB signal

The IFC supports one write protect and one ready/busy signal per chip select. The available write protect and ready/busy signals are mapped to the seven chip selects that the IFC supports.

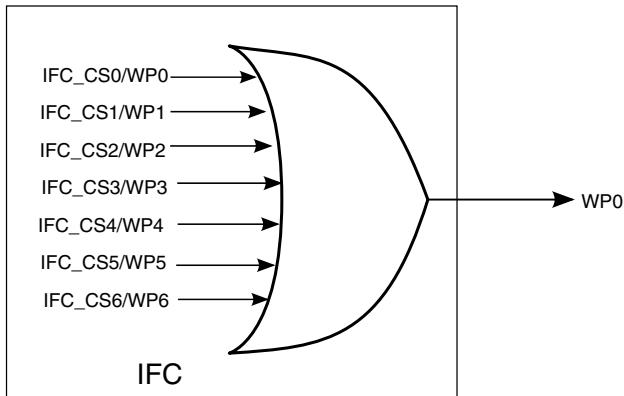
The chip supports one dedicated write protect signal (IFC\_WP0\_B), three multiplexed write protect pins (IFC\_WP[1:3]\_B), one dedicated ready/busy signals (IFC\_RB0\_B) and three multiplexed ready/busy signals IFC\_RB[1:3]\_B .

**NOTE**

The internal connectivity of WP/RB signal depends on the IFC mode selected by RCW[IFC\_MODE].

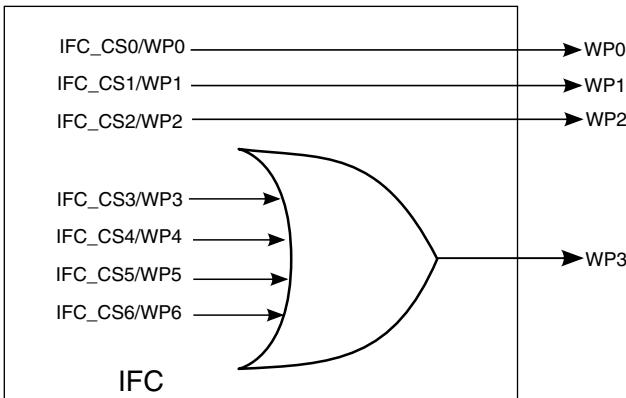
### 25.2.1.1 Internal connectivity of WP signal

For the 25/28-bit address mode, the write protect signal (WP0\_B) is sourced by the logical OR of all the seven write protect signals from the IFC block which corresponds to chip selects 0-6.



**Figure 25-2. Internal connectivity of WP signal for 25/28-bit address mode**

For the 22-bit address mode, the first three write protect signals WP\_B[0:2] are directly sourced by the three write protect signals from the IFC block which correspond to chip selects 0-2, respectively. The fourth write protect signal (WP\_B[3]) is sourced by the logical OR of the remaining four write protect signals from the IFC block which correspond to chip selects 3-6. This allows for some flexibility in controlling write protection to individual or groups of NAND flash devices.

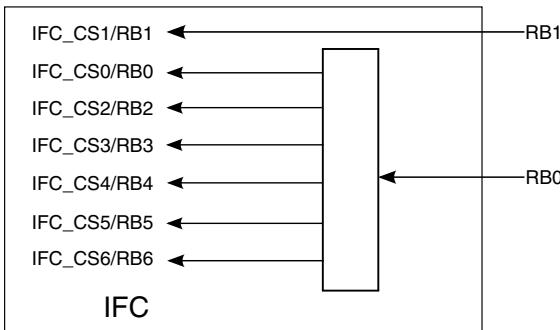


**Figure 25-3. Internal connectivity of WP signal for 22-bit address mode**

### 25.2.1.2 Internal connectivity of RB signal

For the 28-bit address mode:RB1\_B is directly routed to the ready/busy inputs of the IFC block which correspond to chip select 1.

The first ready/busy signal RB0\_B is routed to the ready/busy inputs of the IFC block which correspond to chip selects 0 and 2-6. In these modes of operation, this overall solution allows for one bank of flash to be fully optimized from a performance perspective, while leaving one additional ready/busy for one or more ASICs or lower performance tier of flash devices.



**Figure 25-4. Internal connectivity of RB signal for 28-bit address mode**

For the 22-/25-bit address mode:RB[0:2]\_B is directly routed to the ready/busy inputs of the IFC block which correspond to chip selects 0-2. The fourth ready/busy signal RB3\_B is routed to the ready/busy inputs of the IFC block which correspond to chip selects 3-6 . Note that If IFC\_CS\_B[5] is used, then IFC\_RB\_B[3] cannot be used and vice-versa. This configuration allows for four banks of flash to be fully optimized from a performance perspective, while leaving one additional ready/busy for one or more ASICs or lower performance tier of flash devices.

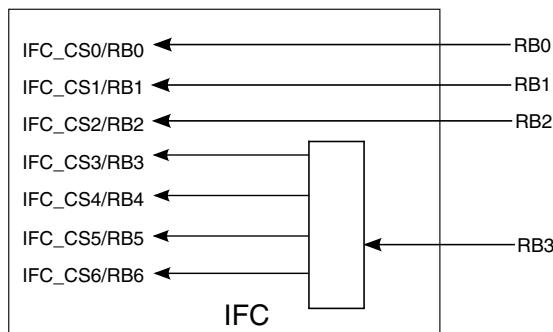


Figure 25-5. Internal connectivity of RB signal for 22/25-bit address mode

## 25.3 IFC memory map/register definition

The IFC is allocated 8 KB of memory-mapped space. The memory map is divided into two parts (4 KB each):

- Common registers shared by NAND, NOR, and GPCM FCM
- Specific registers defined for the NAND FCM, NOR FCM (IFC global), and GPCM FCM (IFC run time) exclusively

IFC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
153_0000	IFC Revision Control register (IFC_REV)	32	R	<a href="#">See section</a>	<a href="#">25.3.1/1215</a>
153_000C	Extended Chip Select Property registers (IFC_CSPR0_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_0010	Chip-select Property register n (IFC_CSPR0)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_0018	Extended Chip Select Property registers (IFC_CSPR1_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_001C	Chip-select Property register n (IFC_CSPR1)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_0024	Extended Chip Select Property registers (IFC_CSPR2_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_0028	Chip-select Property register n (IFC_CSPR2)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_0030	Extended Chip Select Property registers (IFC_CSPR3_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_0034	Chip-select Property register n (IFC_CSPR3)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_003C	Extended Chip Select Property registers (IFC_CSPR4_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_0040	Chip-select Property register n (IFC_CSPR4)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_0048	Extended Chip Select Property registers (IFC_CSPR5_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_004C	Chip-select Property register n (IFC_CSPR5)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_0054	Extended Chip Select Property registers (IFC_CSPR6_EXT)	32	R/W	0000_0000h	<a href="#">25.3.2/1215</a>
153_0058	Chip-select Property register n (IFC_CSPR6)	32	R/W	0000_0000h	<a href="#">25.3.3/1216</a>
153_00A0	Address Mask register (IFC_AMAS0K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
153_00AC	Address Mask register (IFC_AMAS1K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_00B8	Address Mask register (IFC_AMAS2K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_00C4	Address Mask register (IFC_AMAS3K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_00D0	Address Mask register (IFC_AMAS4K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_00DC	Address Mask register (IFC_AMAS5K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_00E8	Address Mask register (IFC_AMAS6K)	32	R/W	0000_0000h	<a href="#">25.3.4/1218</a>
153_0130	Chip-Select Option register - NAND Flash Mode (IFC_CSOR0_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>
153_0130	Chip-Select Option register - NOR Flash Mode (IFC_CSOR0_NOR)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.6/1223</a>
153_0130	Chip-Select Option register - GPCM (IFC_CSOR0_GPCM)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.7/1225</a>
153_0134	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR0_EXT)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.8/1229</a>
153_013C	Chip-Select Option register - NAND Flash Mode (IFC_CSOR1_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>
153_013C	Chip-Select Option register - NOR Flash Mode (IFC_CSOR1_NOR)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.6/1223</a>
153_013C	Chip-Select Option register - GPCM (IFC_CSOR1_GPCM)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.7/1225</a>
153_0140	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR1_EXT)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.8/1229</a>
153_0148	Chip-Select Option register - NAND Flash Mode (IFC_CSOR2_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>
153_0148	Chip-Select Option register - NOR Flash Mode (IFC_CSOR2_NOR)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.6/1223</a>
153_0148	Chip-Select Option register - GPCM (IFC_CSOR2_GPCM)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.7/1225</a>
153_014C	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR2_EXT)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.8/1229</a>
153_0154	Chip-Select Option register - NAND Flash Mode (IFC_CSOR3_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>
153_0154	Chip-Select Option register - NOR Flash Mode (IFC_CSOR3_NOR)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.6/1223</a>
153_0154	Chip-Select Option register - GPCM (IFC_CSOR3_GPCM)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.7/1225</a>
153_0158	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR3_EXT)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.8/1229</a>
153_0160	Chip-Select Option register - NAND Flash Mode (IFC_CSOR4_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>
153_0160	Chip-Select Option register - NOR Flash Mode (IFC_CSOR4_NOR)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.6/1223</a>
153_0160	Chip-Select Option register - GPCM (IFC_CSOR4_GPCM)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.7/1225</a>
153_0164	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR4_EXT)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.8/1229</a>
153_016C	Chip-Select Option register - NAND Flash Mode (IFC_CSOR5_NAND)	32	R/W	<a href="#">See section</a>	<a href="#">25.3.5/1220</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
153_016C	Chip-Select Option register - NOR Flash Mode (IFC_CSOR5_NOR)	32	R/W	<a href="#">See section</a>	25.3.6/1223
153_016C	Chip-Select Option register - GPCM (IFC_CSOR5_GPCM)	32	R/W	<a href="#">See section</a>	25.3.7/1225
153_0170	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR5_EXT)	32	R/W	<a href="#">See section</a>	25.3.8/1229
153_0178	Chip-Select Option register - NAND Flash Mode (IFC_CSOR6_NAND)	32	R/W	<a href="#">See section</a>	25.3.5/1220
153_0178	Chip-Select Option register - NOR Flash Mode (IFC_CSOR6_NOR)	32	R/W	<a href="#">See section</a>	25.3.6/1223
153_0178	Chip-Select Option register - GPCM (IFC_CSOR6_GPCM)	32	R/W	<a href="#">See section</a>	25.3.7/1225
153_017C	Extended Chip-Select Option register - NAND Flash Mode (IFC_CSOR6_EXT)	32	R/W	<a href="#">See section</a>	25.3.8/1229
153_01C0	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS0_NAND)	32	R/W	0000_0000h	25.3.9/1231
153_01C0	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS0_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.10/1233
153_01C0	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS0_NOR)	32	R/W	0000_0000h	25.3.11/1235
153_01C0	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS0_GPCM)	32	R/W	0000_0000h	25.3.12/1236
153_01C4	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS0_NAND)	32	R/W	0000_0000h	25.3.13/1237
153_01C4	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS0_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.14/1238
153_01C4	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS0_NOR)	32	R/W	0000_0000h	25.3.15/1239
153_01C4	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS0_GPCM)	32	R/W	0000_0000h	25.3.16/1240
153_01C8	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS0_NAND)	32	R/W	0000_0000h	25.3.17/1241
153_01C8	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS0_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.18/1241
153_01C8	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS0_NOR)	32	R/W	0000_0000h	25.3.19/1243
153_01C8	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS0_GPCM)	32	R/W	0000_0000h	25.3.20/1244
153_01CC	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS0_NAND)	32	R/W	0000_0000h	25.3.21/1245
153_01CC	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS0_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.22/1246
153_01CC	Flash Timing register 3 for CSn - NOR Flash Mode (IFC_FTIM3_CS_NOR)	32	R/W	0000_0000h	25.3.23/1246

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_01CC	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS0_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.24/ 1247</a>
153_01F0	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS1_NAND)	32	R/W	0000_0000h	<a href="#">25.3.9/1231</a>
153_01F0	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS1_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.10/ 1233</a>
153_01F0	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS1_NOR)	32	R/W	0000_0000h	<a href="#">25.3.11/ 1235</a>
153_01F0	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS1_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.12/ 1236</a>
153_01F4	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS1_NAND)	32	R/W	0000_0000h	<a href="#">25.3.13/ 1237</a>
153_01F4	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS1_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.14/ 1238</a>
153_01F4	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS1_NOR)	32	R/W	0000_0000h	<a href="#">25.3.15/ 1239</a>
153_01F4	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS1_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.16/ 1240</a>
153_01F8	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS1_NAND)	32	R/W	0000_0000h	<a href="#">25.3.17/ 1241</a>
153_01F8	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS1_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.18/ 1241</a>
153_01F8	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS1_NOR)	32	R/W	0000_0000h	<a href="#">25.3.19/ 1243</a>
153_01F8	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS1_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.20/ 1244</a>
153_01FC	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS1_NAND)	32	R/W	0000_0000h	<a href="#">25.3.21/ 1245</a>
153_01FC	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS1_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.22/ 1246</a>
153_01FC	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS1_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.24/ 1247</a>
153_0220	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS2_NAND)	32	R/W	0000_0000h	<a href="#">25.3.9/1231</a>
153_0220	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS2_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.10/ 1233</a>
153_0220	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS2_NOR)	32	R/W	0000_0000h	<a href="#">25.3.11/ 1235</a>
153_0220	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS2_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.12/ 1236</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
153_0224	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS2_NAND)	32	R/W	0000_0000h	<a href="#">25.3.13/ 1237</a>
153_0224	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS2_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.14/ 1238</a>
153_0224	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS2_NOR)	32	R/W	0000_0000h	<a href="#">25.3.15/ 1239</a>
153_0224	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS2_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.16/ 1240</a>
153_0228	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS2_NAND)	32	R/W	0000_0000h	<a href="#">25.3.17/ 1241</a>
153_0228	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS2_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.18/ 1241</a>
153_0228	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS2_NOR)	32	R/W	0000_0000h	<a href="#">25.3.19/ 1243</a>
153_0228	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS2_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.20/ 1244</a>
153_022C	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS2_NAND)	32	R/W	0000_0000h	<a href="#">25.3.21/ 1245</a>
153_022C	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS2_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.22/ 1246</a>
153_022C	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS2_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.24/ 1247</a>
153_0250	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS3_NAND)	32	R/W	0000_0000h	<a href="#">25.3.9/1231</a>
153_0250	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS3_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.10/ 1233</a>
153_0250	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS3_NOR)	32	R/W	0000_0000h	<a href="#">25.3.11/ 1235</a>
153_0250	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS3_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.12/ 1236</a>
153_0254	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS3_NAND)	32	R/W	0000_0000h	<a href="#">25.3.13/ 1237</a>
153_0254	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS3_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.14/ 1238</a>
153_0254	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS3_NOR)	32	R/W	0000_0000h	<a href="#">25.3.15/ 1239</a>
153_0254	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS3_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.16/ 1240</a>
153_0258	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS3_NAND)	32	R/W	0000_0000h	<a href="#">25.3.17/ 1241</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_0258	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS3_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.18/ 1241
153_0258	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS3_NOR)	32	R/W	0000_0000h	25.3.19/ 1243
153_0258	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS3_GPCM)	32	R/W	0000_0000h	25.3.20/ 1244
153_025C	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS3_NAND)	32	R/W	0000_0000h	25.3.21/ 1245
153_025C	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS3_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.22/ 1246
153_025C	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS3_GPCM)	32	R/W	0000_0000h	25.3.24/ 1247
153_0280	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS4_NAND)	32	R/W	0000_0000h	25.3.9/1231
153_0280	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS4_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.10/ 1233
153_0280	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS4_NOR)	32	R/W	0000_0000h	25.3.11/ 1235
153_0280	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS4_GPCM)	32	R/W	0000_0000h	25.3.12/ 1236
153_0284	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS4_NAND)	32	R/W	0000_0000h	25.3.13/ 1237
153_0284	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS4_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.14/ 1238
153_0284	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS4_NOR)	32	R/W	0000_0000h	25.3.15/ 1239
153_0284	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS4_GPCM)	32	R/W	0000_0000h	25.3.16/ 1240
153_0288	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS4_NAND)	32	R/W	0000_0000h	25.3.17/ 1241
153_0288	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS4_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.18/ 1241
153_0288	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS4_NOR)	32	R/W	0000_0000h	25.3.19/ 1243
153_0288	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS4_GPCM)	32	R/W	0000_0000h	25.3.20/ 1244
153_028C	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS4_NAND)	32	R/W	0000_0000h	25.3.21/ 1245
153_028C	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS4_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.22/ 1246

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_028C	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS4_GPCM)	32	R/W	0000_0000h	25.3.24/ 1247
153_02B0	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS5_NAND)	32	R/W	0000_0000h	25.3.9/1231
153_02B0	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS5_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.10/ 1233
153_02B0	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS5_NOR)	32	R/W	0000_0000h	25.3.11/ 1235
153_02B0	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS5_GPCM)	32	R/W	0000_0000h	25.3.12/ 1236
153_02B4	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS5_NAND)	32	R/W	0000_0000h	25.3.13/ 1237
153_02B4	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS5_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.14/ 1238
153_02B4	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS5_NOR)	32	R/W	0000_0000h	25.3.15/ 1239
153_02B4	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS5_GPCM)	32	R/W	0000_0000h	25.3.16/ 1240
153_02B8	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS5_NAND)	32	R/W	0000_0000h	25.3.17/ 1241
153_02B8	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS5_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.18/ 1241
153_02B8	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS5_NOR)	32	R/W	0000_0000h	25.3.19/ 1243
153_02B8	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS5_GPCM)	32	R/W	0000_0000h	25.3.20/ 1244
153_02BC	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS5_NAND)	32	R/W	0000_0000h	25.3.21/ 1245
153_02BC	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS5_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.22/ 1246
153_02BC	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS5_GPCM)	32	R/W	0000_0000h	25.3.24/ 1247
153_02E0	Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode (IFC_FTIM0_CS6_NAND)	32	R/W	0000_0000h	25.3.9/1231
153_02E0	Flash Timing register 0 for Chip Select n - NAND flash Asynchronous Mode (IFC_FTIM0_CS6_NAND_ASYNC_MODE)	32	R/W	0000_0000h	25.3.10/ 1233
153_02E0	Flash Timing register 0 for CSn - NOR Flash Mode (IFC_FTIM0_CS6_NOR)	32	R/W	0000_0000h	25.3.11/ 1235
153_02E0	Flash Timing register 0 for CSn - Normal GPCM Mode (IFC_FTIM0_CS6_GPCM)	32	R/W	0000_0000h	25.3.12/ 1236

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_02E4	Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC_FTIM1_CS6_NAND)	32	R/W	0000_0000h	<a href="#">25.3.13/ 1237</a>
153_02E4	Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC_FTIM1_CS6_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.14/ 1238</a>
153_02E4	Flash Timing register 1 for CSn - NOR Flash Mode (IFC_FTIM1_CS6_NOR)	32	R/W	0000_0000h	<a href="#">25.3.15/ 1239</a>
153_02E4	Flash Timing register 1 for CSn - Normal GPCM Mode (IFC_FTIM1_CS6_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.16/ 1240</a>
153_02E8	Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC_FTIM2_CS6_NAND)	32	R/W	0000_0000h	<a href="#">25.3.17/ 1241</a>
153_02E8	Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM2_CS6_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.18/ 1241</a>
153_02E8	Flash Timing register 2 for CSn - NOR Flash Mode (IFC_FTIM2_CS6_NOR)	32	R/W	0000_0000h	<a href="#">25.3.19/ 1243</a>
153_02E8	Flash Timing register 2 for CSn - Normal GPCM Mode (IFC_FTIM2_CS6_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.20/ 1244</a>
153_02EC	Flash Timing register 3 for Chip Select n - NAND Flash Mode (IFC_FTIM3_CS6_NAND)	32	R/W	0000_0000h	<a href="#">25.3.21/ 1245</a>
153_02EC	Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC_FTIM3_CS6_NAND_ASYNC_MODE)	32	R/W	0000_0000h	<a href="#">25.3.22/ 1246</a>
153_02EC	Flash Timing register 3 for CSn - Normal GPCM Mode (IFC_FTIM3_CS6_GPCM)	32	R/W	0000_0000h	<a href="#">25.3.24/ 1247</a>
153_0400	Ready Busy Status for each Chip Select (IFC_RB_STAT)	32	R	<a href="#">See section</a>	<a href="#">25.3.25/ 1248</a>
153_040C	General Control register (IFC_GCR)	32	R/W	0000_0000h	<a href="#">25.3.26/ 1249</a>
153_0418	Common Event and Error Status register (IFC_CM_EVTER_STAT)	32	w1c	0000_0000h	<a href="#">25.3.27/ 1250</a>
153_0424	Common Event and Error Enable register (IFC_CM_EVTER_EN)	32	R/W	8000_0000h	<a href="#">25.3.28/ 1251</a>
153_0430	Common Event and Error Interrupt Enable register (IFC_CM_EVTER_INTR_EN)	32	R/W	0000_0000h	<a href="#">25.3.29/ 1252</a>
153_043C	Common Transfer Error Attributes register 0 (IFC_CM_ERATTR0)	32	R	0000_0000h	<a href="#">25.3.30/ 1252</a>
153_0440	Common Transfer Error Attributes register 1 (IFC_CM_ERATTR1)	32	R	0000_0000h	<a href="#">25.3.31/ 1254</a>
153_044C	Clock Control register (IFC_CCR)	32	R/W	0300_8000h	<a href="#">25.3.32/ 1254</a>
153_0450	Clock Status register (IFC_CSR)	32	R	<a href="#">See section</a>	<a href="#">25.3.33/ 1256</a>
153_0454	DDR Clock Control register (IFC_DDR_CCR)	32	R/W	0080_0000h	<a href="#">25.3.34/ 1257</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_1000	NAND Configuration register (IFC_NCFGR)	32	R/W	0000_0000h	<a href="#">25.3.35/ 1259</a>
153_1014	NAND Flash Command register 0 (IFC_NAND_FCR0)	32	R/W	0000_0000h	<a href="#">25.3.36/ 1261</a>
153_1018	NAND Flash Command register 1 (IFC_NAND_FCR1)	32	R/W	0000_0000h	<a href="#">25.3.37/ 1262</a>
153_103C	Flash Row Address register n (IFC_ROW0)	32	R/W	0000_0000h	<a href="#">25.3.38/ 1262</a>
153_1044	Flash COL Address register n (IFC_COL0)	32	R/W	0000_0000h	<a href="#">25.3.39/ 1263</a>
153_1044	Flash COL Address register for 2 KB Large-Page Device (IFC_COL0_2KB)	32	R/W	0000_0000h	<a href="#">25.3.40/ 1264</a>
153_1044	Flash COL Address register for 4 KB Large-Page Device (IFC_COL0_4KB)	32	R/W	0000_0000h	<a href="#">25.3.41/ 1265</a>
153_1044	Flash COL Address register for 8 KB Large-Page Device (IFC_COL0_8KB)	32	R/W	0000_0000h	<a href="#">25.3.42/ 1266</a>
153_104C	Flash Row Address register n (IFC_ROW1)	32	R/W	0000_0000h	<a href="#">25.3.38/ 1262</a>
153_1054	Flash COL Address register n (IFC_COL1)	32	R/W	0000_0000h	<a href="#">25.3.39/ 1263</a>
153_1054	Flash COL Address register for 2 KB Large-Page Device (IFC_COL1_2KB)	32	R/W	0000_0000h	<a href="#">25.3.40/ 1264</a>
153_1054	Flash COL Address register for 4 KB Large-Page Device (IFC_COL1_4KB)	32	R/W	0000_0000h	<a href="#">25.3.41/ 1265</a>
153_1054	Flash COL Address register for 8 KB Large-Page Device (IFC_COL1_8KB)	32	R/W	0000_0000h	<a href="#">25.3.42/ 1266</a>
153_105C	Flash Row Address register n (IFC_ROW2)	32	R/W	0000_0000h	<a href="#">25.3.38/ 1262</a>
153_1064	Flash COL Address register n (IFC_COL2)	32	R/W	0000_0000h	<a href="#">25.3.39/ 1263</a>
153_1064	Flash COL Address register for 2 KB Large-Page Device (IFC_COL2_2KB)	32	R/W	0000_0000h	<a href="#">25.3.40/ 1264</a>
153_1064	Flash COL Address register for 4 KB Large-Page Device (IFC_COL2_4KB)	32	R/W	0000_0000h	<a href="#">25.3.41/ 1265</a>
153_1064	Flash COL Address register for 8 KB Large-Page Device (IFC_COL2_8KB)	32	R/W	0000_0000h	<a href="#">25.3.42/ 1266</a>
153_106C	Flash Row Address register n (IFC_ROW3)	32	R/W	0000_0000h	<a href="#">25.3.38/ 1262</a>
153_1074	Flash COL Address register n (IFC_COL3)	32	R/W	0000_0000h	<a href="#">25.3.39/ 1263</a>
153_1074	Flash COL Address register for 2 KB Large-Page Device (IFC_COL3_2KB)	32	R/W	0000_0000h	<a href="#">25.3.40/ 1264</a>
153_1074	Flash COL Address register for 4 KB Large-Page Device (IFC_COL3_4KB)	32	R/W	0000_0000h	<a href="#">25.3.41/ 1265</a>

Table continues on the next page...

**IFC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_1074	Flash COL Address register for 8 KB Large-Page Device (IFC_COL3_8KB)	32	R/W	0000_0000h	<a href="#">25.3.42/ 1266</a>
153_1108	Flash Byte Count register for NAND Flash (IFC_NAND_BC)	32	R/W	0000_0000h	<a href="#">25.3.43/ 1267</a>
153_1110	NAND Flash Instruction register 0 (IFC_NAND_FIR0)	32	R/W	0000_0000h	<a href="#">25.3.44/ 1268</a>
153_1114	NAND Flash Instruction register 1 (IFC_NAND_FIR1)	32	R/W	0000_0000h	<a href="#">25.3.45/ 1270</a>
153_1118	NAND Flash Instruction register 2 (IFC_NAND_FIR2)	32	R/W	0000_0000h	<a href="#">25.3.46/ 1271</a>
153_115C	NAND Chip-Select register (IFC_NAND_CSEL)	32	R/W	0000_0000h	<a href="#">25.3.47/ 1272</a>
153_1164	NAND Operation Sequence Start (IFC_NANDSEQ_STRT)	32	R/W	0000_0000h	<a href="#">25.3.48/ 1272</a>
153_116C	NAND Event and Error Status register (IFC_NAND_EVTER_STAT)	32	w1c	0000_0000h	<a href="#">25.3.49/ 1274</a>
153_1174	NAND Page Read Completion Event Status register (IFC_PGRDCMPL_EVT_STAT)	32	w1c	0000_0000h	<a href="#">25.3.50/ 1276</a>
153_1180	NAND Event and Error Enable register (IFC_NAND_EVTER_EN)	32	R/W	AE00_0000h	<a href="#">25.3.51/ 1278</a>
153_118C	NAND Event and Error Interrupt Enable register (IFC_NAND_EVTER_INTR_EN)	32	R/W	0000_0000h	<a href="#">25.3.52/ 1280</a>
153_1198	NAND Transfer Error Attributes register 0 (IFC_NAND_ERATTR0)	32	R	0000_0000h	<a href="#">25.3.53/ 1281</a>
153_119C	NAND Transfer Error Attributes register 1 (IFC_NAND_ERATTR1)	32	R	0000_0000h	<a href="#">25.3.54/ 1282</a>
153_11E0	NAND Flash Status register (IFC_NAND_FSR)	32	R	0000_0000h	<a href="#">25.3.55/ 1283</a>
153_11E8	ECC Status and Result of Flash Operation register 0 (IFC_ECCSTAT0)	32	R	0000_0000h	<a href="#">25.3.56/ 1283</a>
153_11EC	ECC Status and Result of Flash Operation register 1 (IFC_ECCSTAT1)	32	R	0000_0000h	<a href="#">25.3.57/ 1285</a>
153_11F0	ECC Status and Result of Flash Operation register 2 (IFC_ECCSTAT2)	32	R	0000_0000h	<a href="#">25.3.58/ 1286</a>
153_11F4	ECC Status and Result of Flash Operation register 3 (IFC_ECCSTAT3)	32	R	0000_0000h	<a href="#">25.3.59/ 1287</a>
153_1278	NAND Control register (IFC_NANDCR)	32	R/W	1E00_0000h	<a href="#">25.3.60/ 1288</a>
153_1284	NAND Autoboot Trigger register (IFC_NAND_AUTOBOOT_TRGR)	32	W	0000_0000h	<a href="#">25.3.61/ 1288</a>
153_128C	NAND Flash Memory Data register (IFC_NAND_MDR)	32	R	0000_0000h	<a href="#">25.3.62/ 1290</a>
153_1300	Nand DLL Low Config 0 Register (IFC_NAND_DLL_LOW_CFG0)	32	R/W	8007_0000h	<a href="#">25.3.63/ 1291</a>

Table continues on the next page...

## IFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
153_1304	Nand DLL Low Config 1 Register (IFC_NAND_DLL_LOW_CFG1)	32	R/W	8000_0000h	<a href="#">25.3.64/ 1292</a>
153_130C	NAND DLL Low Status Register (IFC_NAND_DLL_LOW_STAT)	32	R	0000_0000h	<a href="#">25.3.65/ 1294</a>
153_1400	NOR Event and Error Status register (IFC_NOR_EVTER_STAT)	32	w1c	0000_0000h	<a href="#">25.3.66/ 1295</a>
153_140C	NOR Event and Error Enable register (IFC_NOR_EVTER_EN)	32	R/W	8500_0000h	<a href="#">25.3.67/ 1297</a>
153_1418	NOR Event and Error Interrupt enable register (IFC_NOR_EVTER_INTR_EN)	32	R/W	0000_0000h	<a href="#">25.3.68/ 1299</a>
153_1424	NOR Transfer Error Attributes register 0 (IFC_NOR_ERATTR0)	32	R	0000_0000h	<a href="#">25.3.69/ 1300</a>
153_1428	NOR Transfer Error Attribute register 1 (IFC_NOR_ERATTR1)	32	R	0000_0000h	<a href="#">25.3.70/ 1301</a>
153_142C	NOR Transfer Error Attribute register 2 (IFC_NOR_ERATTR2)	32	R	0000_0000h	<a href="#">25.3.71/ 1302</a>
153_1440	NOR Control register (IFC_NORCR)	32	R/W	000F_0000h	<a href="#">25.3.72/ 1302</a>
153_1800	GPCM Event and Error Status register (IFC_GPCM_EVTER_STAT)	32	w1c	0000_0000h	<a href="#">25.3.73/ 1304</a>
153_180C	GPCM Event and Error Enable register (IFC_GPCM_EVTER_EN)	32	R/W	0540_0000h	<a href="#">25.3.74/ 1306</a>
153_1818	GPCM Event and Error Interrupt enable register (IFC_GPCM_EVTER_INTR_EN)	32	R/W	0000_0000h	<a href="#">25.3.75/ 1307</a>
153_1824	GPCM Transfer Error Attributes register 0 (IFC_GPCM_ERATTR0)	32	R	0000_0000h	<a href="#">25.3.76/ 1309</a>
153_1828	GPCM Transfer Error Attributes register 1 (IFC_GPCM_ERATTR1)	32	R	0000_0000h	<a href="#">25.3.77/ 1310</a>
153_182C	GPCM Transfer Error Attributes register 2 (IFC_GPCM_ERATTR2)	32	R	0000_0000h	<a href="#">25.3.78/ 1311</a>
153_1830	GPCM Status register (IFC_GPCM_STAT)	32	R	0000_0000h	<a href="#">25.3.79/ 1313</a>

### 25.3.1 IFC Revision Control register (IFC\_REV)

This register represents the revision number of the IFC.

Address: 153\_0000h base + 0h offset = 153\_0000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				REV_MAJ		Reserved			REV_MIN		Reserved																				
W																																
Reset	0	0	0	0	0	0	n	n	0	0	0	0	n	n	n	n	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### IFC\_REV field descriptions

Field	Description
0–3 -	This field is reserved.
4–7 REV_MAJ	Major Revision. It represents the major revision of IFC
8–11 -	This field is reserved.
12–15 REV_MIN	Minor Revision. It represents the minor revision of IFC.
16–31 -	This field is reserved.

### 25.3.2 Extended Chip Select Property registers (IFC\_CSPRn\_EXT)

The extended chip select property register (CSPRn\_EXT) contains the extended base address, that is, the most significant bits (msb) of the base address.

Address: 153\_0000h base + Ch offset + (12d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															BA_EXT																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### IFC\_CSPRn\_EXT field descriptions

Field	Description
0–23 -	This field is reserved.

Table continues on the next page...

**IFC\_CSPRn\_EXT field descriptions (continued)**

Field	Description
24–31 BA_EXT	Extended Base Address: This field contains the msbs of the base address. Complete base address can be represented by concatenating CSPRn_EXT[BA_EXT] and CSPRn[BA] fields. Each chip select's base register value is compared to the address on the address bus to determine if the master is accessing a memory bank controlled by the IFC. Address decoding is performed by using the address mask bit in AMASKn[AM] field more details of decoding is given in AMASK register description.

**25.3.3 Chip-select Property register n (IFC\_CSPRn)**

The chip-select property register (CSPRn) contains the base address and memory attributes for each bank.

**NOTE**

CSPR0: Only chip-select 0 is used for booting. PS, MSEL[1], and TE is obtained from the configuration word when boot\_load/rcw\_load occurs. V will be set for CSPR0 when boot\_load/rcw\_load occurs with valid port size.

Address: 153\_0000h base + 10h offset + (12d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_CSPRn field descriptions**

Field	Description
0–15 BA	Base Address: Each base register value is compared to the address on the address bus to determine if the master is accessing a memory bank controlled by the IFC. Used with the address mask bit.
16–22 -	This field is reserved.
23–24 PS	Port Size-Specifies the port size of this memory region. For CSPR0, PS is configured through reset configuration word as loaded during power on reset. For all other banks the value is reset to 00 (port size not defined). 00 Reserved 01 8 bit 10 16 bit 11 Reserved
25 WP	Write Protect:  <b>NOTE:</b> 1. This bit is valid only for NAND and NOR; for GPCM this bit should not be set.  <b>NOTE:</b> 2. If CS0 is used for booting from NAND or NOR, CSPR0[WP] will be set (write protected). Software must clear this bit after completing the boot operation to permit write operations on CS0 0 Read and write accesses are allowed. 1 Only read accesses are allowed. The memory controller does not assert chip-select on write cycles to this memory bank for NOR flash. (Refer <a href="#">Write protect</a> for more details)
26 -	This field is reserved.
27 TE	External Transceiver Enable. It specifies the value that will be driven on TE pin when a particular CSn is selected. 0 Logic 0 will be driven on TE pin 1 Logic 1 will be driven on TE pin
28 -	This field is reserved.
29–30 MSEL	Machine Select 00 NOR flash 01 NAND flash 10 GPCM 11 Reserved
31 V	Valid: Indicates that the contents of CSPRn are valid. 1 Bank is valid 0 Bank is invalid

### 25.3.4 Address Mask register (IFC\_AMASnK)

The address mask (AMASKn) registers contains the 16-bit address mask field for all four memory banks. The selected 16-bit address mask field masks corresponding CSPRn[BA] fields. The 16 lsbs of the 32-bit internal address do not participate in bank address matching in selecting a bank for access. Masking address bits independently allows external devices of different size address ranges to be used. Address mask bits can be set or cleared in any order in the field, allowing a resource to reside in more than one area of the address map.

The IFC only has 32 address pins at the external memory interface side. Since the system side address bus width is 40 bits, only the lower 32 bits will be given out. The upper 8 bits are used only for address decoding to identify the bank (chip select) on which access will be performed. There is no masking of the upper 8 bits of the system side address so it will be compared with CSPRn\_EXT[BA\_EXT] field as it is. Therefore, if the user programs the same value in the CSPRn\_EXT[BA\_EXT] field, the entire address range of the IFC will be 4 GB. If different values are programmed in the CSPRn\_EXT[BA\_EXT] field, the address range of the IFC will then be (number of chip selects) x 4 GB, that is, 32 GB.

The following logic is used for address decoding:

24-bit base address is formed as,  $\text{BASE\_ADDRn}[0:23] = \{\text{CSPRn\_EXT}[BA\_EXT], \text{CSPRn}[BA]\}$ ,  $n=0-6$  (chip select)

Select CSn (chip select n) if,

$\{\text{BASE\_ADDRn}[0:7], (\text{BASE\_ADDRn}[8:23] \& \text{AMn}[0:15])\} == \{\text{SYSTEM\_ADDR}[39:32], (\text{SYSTEM\_ADDR}[31:16] \& \text{AMn}[0:15])\}$ , where

- SYSTEM\_ADDR is a 40-bit incoming address from the system side
- Index 39-32 represents 8 address msbs
- SYSTEM\_ADDR[31:16] represent the next lower 16 address bits and the remaining 16 lsbs are not used in the bank selection
- In the logic mentioned above, the 8 msbs are not masked and are used as is for comparison.

#### NOTE

Chip select 0 (CS0) is used for boot purposes; if the boot source is NOR, then it must be executed in place. During booting, the IFC registers (including BA\_EXT, BA, and AMASK registers) are modified. If there is race condition between the update of BA\_EXT, BA, and AMASK registers, a chip select error may

occur. To avoid this error and to map every transaction in CS0 during boot, the upper 8 bits of the system address and the base address, will be masked before comparison. Masking of these fields is turned off only after the first write transaction modifies the AMASK0 register.

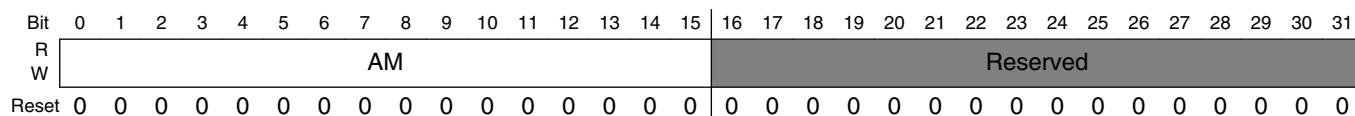
After reset, until a first write transaction occurs to update AMASK0 register, all the transactions coming from the system side will always be mapped to CS0. After receiving the first write transaction to update AMASK0, the chip select decoding logic will work as per the above mentioned equation.

The table below shows memory bank size from 64 KB to 4 GB.

**Table 25-3. Memory Bank Sizes in Relation to Address Mask**

AM	Memory Bank Size
0000_0000_0000_0000	4 GBytes
1000_0000_0000_0000	2 Gbytes
1100_0000_0000_0000	1 Gbytes
1110_0000_0000_0000	512 MB
1111_0000_0000_0000	256 MB
1111_1000_0000_0000	128 MB
1111_1100_0000_0000	64 MB
1111_1110_0000_0000	32 MB
1111_1111_0000_0000	16 MB
1111_1111_1000_0000	8 MB
1111_1111_1100_0000	4 MB
1111_1111_1110_0000	2 MB
1111_1111_1111_0000	1 MB
1111_1111_1111_1000	512 KB
1111_1111_1111_1100	256 KB
1111_1111_1111_1110	128 KB
1111_1111_1111_1111	64 KB

Address: 153\_0000h base + A0h offset + (12d × i), where i=0d to 6d



### IFC\_AMASnK field descriptions

Field	Description
0–15 AM	16-bit Address mask corresponding to memory bank

*Table continues on the next page...*

## IFC\_AMASnK field descriptions (continued)

Field	Description
16–31 -	This field is reserved.

### 25.3.5 Chip-Select Option register - NAND Flash Mode (IFC\_CSORn\_NAND)

The following figure shows the CSORn fields when CSPRn[MSEL] selects the NAND flash mode.

#### NOTE

- CSOR0 is the only chip-select used for booting.
- ECC\_DEC\_EN, ECC\_MODE[0:1], and PGS are obtained from the configuration word when boot\_load/rcw\_load occurs.

Address: 153\_0000h base + 130h offset + (12d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ECC_ENC_EN	Reserved		ECC_MODE	Reserved	ECC_DEC_EN	Reserved	RAL		Reserved		PGS		Reserved		
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R		SPRZ		Reserved		PB		Reserved		NAND_MODE		TRHZ		Reserved		BCTL
W	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

### IFC\_CSORn\_NAND field descriptions

Field	Description
0 ECC_ENC_EN	ECC Encoder Enable/Disable bit:

Table continues on the next page...

**IFC\_CSORn\_NAND field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> The encoder should be enabled only when performing full page operations. In case of partial page operations, the encoder should be disabled.</p> <p>0 ECC encoding disabled 1 ECC Encoding enabled</p>
1 -	This field is reserved.
2–3 ECC_MODE	<p>ECC Mode of operation</p> <p>00 4 bit correction per 512 byte data sector 01 8 bit correction per 512 byte data sector 10 24 bit correction per 1 KB sector 11 40 bit correction per 1 KB sector</p>
4 -	This field is reserved.
5 ECC_DEC_EN	<p>ECC Decoding Enable/Disable bit</p> <p><b>NOTE:</b> The decoder should be enabled only when performing full page operations. In case of partial page operations, the decoder should be disabled</p> <p>0 ECC decoding disabled 1 ECC decoding enabled</p>
6 -	This field is reserved.
7–8 RAL	<p>Row Address Length: Number of address bytes issued during page address operation</p> <p><b>NOTE:</b> Bytes for column address are determined by page size</p> <p><b>NOTE:</b> For boot time RAL value, see <a href="#">NAND asynchronous mode boot mechanism</a> (the default is set to max assuming that the device will ignore extra address cycles).</p> <p>00 1 byte 01 2 bytes 10 3 bytes 11 4 bytes</p>
9–10 -	This field is reserved.
11–12 PGS	<p>Page Size</p> <p>00 512 Bytes 01 2 KB 10 4 KB 11 8 KB</p>
13–15 -	This field is reserved.
16–18 SPRZ	<p>Spare size</p> <p><b>NOTE:</b> Depending on the ECC mode and page size, a fixed value of the spare region is selected during boot. For more information, see <a href="#">Booting methods</a>.</p> <p>Others Reserved</p>

*Table continues on the next page...*

**IFC\_CSORn\_NAND field descriptions (continued)**

Field	Description
	<p>000 16 Bytes      001 64 Bytes      010 128 Bytes      011 210 Bytes      100 218 Bytes      101 224 Bytes      110 Spare size information will be used from corresponding CSORn_EXT register.</p>
19–20 -	This field is reserved.
21–23 PB	<p>Pages per Block Others Reserved</p> <p>000 32 Pages      001 64 Pages      010 128 Pages      011 256 Pages      100 512 Pages</p>
24 -	This field is reserved.
25–26 NAND_MODE	<p>NAND mode of operation Others Reserved</p> <p>00 Asynchronous mode      01 NVDDR      10 Reserved      11 Reserved</p>
27–29 TRHZ	<p>Time for read enable high to output high impedance (Z). Number of clocks required for memory to go in high-Z after read enable deassertion.</p> <p>This field is used during last read data access. If the IFC is accessing the NAND flash for a read operation, then after the last byte is read the NAND FSM must wait for TRHZ clock cycles so that there is no contention on the external buffer.</p> <p>Other settings are Reserved.</p> <p>000 Wait for 20 IP Clocks      001 Wait for 40 IP Clocks      010 Wait for 60 IP Clocks      011 Wait for 80 IP Clocks      100 Wait for 100 IP Clocks</p>
30 -	This field is reserved.
31 BCTL_D	<p>Buffer control disable. This bit signifies presence or absence of external buffer. If buffer is absent then this bit should be set to 1.</p> <p>0 BCTL is actively driven by IFC based on direction of access.      1 BCTL is set to its default value (high) irrespective of direction of access.</p>

### 25.3.6 Chip-Select Option register - NOR Flash Mode (IFC\_CSORn\_NOR)

The following figure shows the CSORn fields when CSPRn[MSEL] selects the NOR flash mode.

#### NOTE

- CSOR0 is the only chip-select used for booting.
- ADM\_SHFT\_MODE/ADM\_SHFT is obtained from the configuration word when boot\_load/rcw\_load occurs.

Address: 153\_0000h base + 130h offset + (12d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ADM_SHFT_MODE	Reserved		PGRD_EN	Reserved		AVD_TGL_PGM_EN		Reserved						ADM_SHFT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ADM_SHFT			Reserved					NOR_MODE		TRHZ			Reserved		BCTLID
W																
Reset	n	n	n	0	0	0	0	0	0	0	0	0	1	1	0	0

#### IFC\_CSORn\_NOR field descriptions

Field	Description
0 ADM_SHFT_MODE	<p>Address shift mode</p> <p><b>NOTE:</b> Details of shifting is given in <a href="#">Mode 0 pin muxing (CSORn[ADM_SHFT_MODE] = 0)</a>.</p> <p>0 Address msbs will be assigned to AD bus and ADDR bus carries the lsb</p> <p>1 AD bus will carry lsbs and ADDR bus carries the msb</p>
1–2 -	This field is reserved.
3 PGRD_EN	Page read enable from NOR device

Table continues on the next page...

**IFC\_CSORn\_NOR field descriptions (continued)**

Field	Description																																										
	<p><b>NOTE:</b> Software should issue transaction based on device page size and it is valid only for parallel NOR devices and should not be set for ADM NOR devices.</p> <ul style="list-style-type: none"> <li>0 A multi-beat read transaction received from system bus will be split into per-beat accesses (based on NOR port size).</li> <li>1 A multi-beat read transaction received from system bus will be performed as a single-page read operation (burst type) on NOR flash.</li> </ul>																																										
4–6 -	This field is reserved.																																										
7 AVD_TGL_ PGM_EN	<p>AVD toggle enable during burst program</p> <ul style="list-style-type: none"> <li>0 Assert AVD only for the first address phase (performed on the flash interface) of a burst write operation received from the system interface</li> <li>1 Assert AVD during every subsequent address phase (performed on the flash interface) including the first phase of a burst write operation received from the system interface</li> </ul>																																										
8–13 -	This field is reserved.																																										
14–18 ADM_SHFT	<p>Address data multiplexing shift. It controls the way internal 32 bit address is placed on the external bus, during NOR/GPCM address phase. Address shifting will be done by 1 bit.</p> <p>Patterns not shown are reserved.</p> <p>During reset, in case the RCW source is from NOR then,</p> <p>if cfg_rcw_src[6:7]=00 (00 22b addressability) then ADM_SHFT = 10      if cfg_rcw_src[6:7]=01 (00 25b addressability) then ADM_SHFT = 7      if cfg_rcw_src[6:7]=10 (00 28b addressability) then ADM_SHFT = 4</p> <table> <tbody> <tr><td>00000</td><td>No shift</td></tr> <tr><td>00001</td><td>shift ifc_addr by 1</td></tr> <tr><td>00010</td><td>shift ifc_addr by 2</td></tr> <tr><td>00011</td><td>shift ifc_addr by 3</td></tr> <tr><td>00100</td><td>shift ifc_addr by 4</td></tr> <tr><td>00101</td><td>shift ifc_addr by 5</td></tr> <tr><td>00110</td><td>shift ifc_addr by 6</td></tr> <tr><td>00111</td><td>shift ifc_addr by 7</td></tr> <tr><td>01000</td><td>shift ifc_addr by 8</td></tr> <tr><td>01001</td><td>shift ifc_addr by 9</td></tr> <tr><td>01010</td><td>shift ifc_addr by 10</td></tr> <tr><td>01011</td><td>shift ifc_addr by 11</td></tr> <tr><td>01100</td><td>shift ifc_addr by 12</td></tr> <tr><td>01101</td><td>shift ifc_addr by 13</td></tr> <tr><td>01110</td><td>shift ifc_addr by 14</td></tr> <tr><td>01111</td><td>shift ifc_addr by 15</td></tr> <tr><td>10000</td><td>shift ifc_addr by 16</td></tr> <tr><td>10001</td><td>shift ifc_addr by 17</td></tr> <tr><td>10010</td><td>shift ifc_addr by 18</td></tr> <tr><td>10011</td><td>shift ifc_addr by 19</td></tr> <tr><td>10100</td><td>shift ifc_addr by 20</td></tr> </tbody> </table>	00000	No shift	00001	shift ifc_addr by 1	00010	shift ifc_addr by 2	00011	shift ifc_addr by 3	00100	shift ifc_addr by 4	00101	shift ifc_addr by 5	00110	shift ifc_addr by 6	00111	shift ifc_addr by 7	01000	shift ifc_addr by 8	01001	shift ifc_addr by 9	01010	shift ifc_addr by 10	01011	shift ifc_addr by 11	01100	shift ifc_addr by 12	01101	shift ifc_addr by 13	01110	shift ifc_addr by 14	01111	shift ifc_addr by 15	10000	shift ifc_addr by 16	10001	shift ifc_addr by 17	10010	shift ifc_addr by 18	10011	shift ifc_addr by 19	10100	shift ifc_addr by 20
00000	No shift																																										
00001	shift ifc_addr by 1																																										
00010	shift ifc_addr by 2																																										
00011	shift ifc_addr by 3																																										
00100	shift ifc_addr by 4																																										
00101	shift ifc_addr by 5																																										
00110	shift ifc_addr by 6																																										
00111	shift ifc_addr by 7																																										
01000	shift ifc_addr by 8																																										
01001	shift ifc_addr by 9																																										
01010	shift ifc_addr by 10																																										
01011	shift ifc_addr by 11																																										
01100	shift ifc_addr by 12																																										
01101	shift ifc_addr by 13																																										
01110	shift ifc_addr by 14																																										
01111	shift ifc_addr by 15																																										
10000	shift ifc_addr by 16																																										
10001	shift ifc_addr by 17																																										
10010	shift ifc_addr by 18																																										
10011	shift ifc_addr by 19																																										
10100	shift ifc_addr by 20																																										

*Table continues on the next page...*

**IFC\_CSORn\_NOR field descriptions (continued)**

Field	Description
19–24 -	This field is reserved.
25–26 NOR_MODE	Type of the NOR device hooked at CSn. 00 Simple asynchronous NOR (ALE is asserted before CE_B) 01 Internal latch based AVD NOR device (ALE is asserted after CE_B) Others Reserved
27–29 TRHZ	Time for read enable high to output high impedance (Z). Number of clocks required for memory to go in high Z after read enable deassertion. If IFC is accessing NOR flash for read operation, then after last byte read NOR FSM must wait for these many clock cycles so that there is no contention on external buffer. Others Reserved 000 Wait for 20 IP Clocks 001 Wait for 40 IP Clocks 010 Wait for 60 IP Clocks 011 Wait for 80 IP Clocks 100 Wait for 100 IP Clocks
30 -	This field is reserved.
31 BCTL	Buffer control disable. This bit signifies presence or absence of external buffer. If buffer is absent then this bit should be set to 1. 0 BCTL is actively driven by IFC based on direction of access. 1 BCTL is set to its default value (high) irrespective of direction of access.

**25.3.7 Chip-Select Option register - GPCM (IFC\_CSORn\_GPCM)**

The CSCORn can work in following two modes:

- Normal GPCM
- Generic ASIC

The following figure shows the CSORn fields when CSPRn[MSEL] selects the GPCM mode.

## IFC memory map/register definition

Address: 153\_0000h base + 130h offset + (12d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	GPMODE		PAR	PAR_EN	Reserved								ABRT_RSP_EN			
W													RGETA		WGETA	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				ADM_SHFT	Reserved		BURST_LEN		GAPERRD	Reserved		TRHZ			Reserved	
W																BCTLD
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

### IFC\_CSORn\_GPCM field descriptions

Field	Description
0 GPMODE	GPCM Mode of operation: 0 Normal GPCM operation 1 Generic ASIC mode operation
1 PAR	Parity mode. 0 Odd Parity 1 Even Parity
2 PAR_EN	Parity checking enable/disable: 0 Parity checking disabled over the received data 1 Parity checking enabled over the received data
3 -	This field is reserved.
4–7 GPTO	GPCM Timeout Count: For normal GPCM, the timeout occur only when read data/write data transaction is external transaction acknowledgement based and IFCTA signal has not come.  In Generic ASIC mode timeout occurs when RDY_L has not come for the number of IP clks cycles defined by this register field.  0000 256 cycles of IFC module input clocks 0001 512 cycles of IFC module input clocks 0010 1024 cycles of IFC module input clocks 0011 2048 cycles of IFC module input clocks 0100 4096 cycles of IFC module input clocks

Table continues on the next page...

**IFC\_CSORn\_GPCM field descriptions (continued)**

Field	Description
	0101 8192 cycles of IFC module input clocks 0110 16,384 cycles of IFC module input clocks 0111 32,768 cycles of IFC module input clocks 1000 65,536 cycles of IFC module input clocks 1001 131,072 cycles of IFC module input clocks 1010 262,144 cycles of IFC module input clocks 1011 524,288 cycles of IFC module input clocks 1100 1,048,576 cycles of IFC module input clocks 1101 2,097,152 cycles of IFC module input clocks 1110 4,194,304 cycles of IFC module input clocks 1111 8,388,608 cycles of IFC module input clocks
8–10 -	This field is reserved.
11 ABRT_RSP_EN	<p>Abort Error Response Enable. An error response will be sent for read transaction if the transaction is aborted by IFCTA_B when REGTA is programmed in abort mode.</p> <p>This bit is valid only for normal GPCM mode.</p> <p>0 No error response will be sent. 1 Error response will be sent for abort.</p>
12 RGETA	<p>GPCM external access termination mode for read access.</p> <p><b>NOTE:</b> This bit is valid only for normal GPCM Mode.</p> <p>0 Abort mode. IFCTA_B signal acts as abort signal. GPCM read access is terminated internally by the controller at the expiry of TRAD counter unless aborted externally (If IFCTA_B asserts before the expiry of TRAD counter access will be terminated). Error will be reported in GPCM_EVTER_STAT[ABER] register.</p> <p>1 Acknowledgement mode. IFCTA_B acts as acknowledgement/data qualifier signal. GPCM read access is acknowledged by external pin IFCTA_B and only it can complete the read access. If it is not asserted within CSOR[GPTO] time, GPCM_EVTER_STAT[TOER] will be set.</p>
13 WGETA	<p>GPCM external access termination mode for write access:</p> <p><b>NOTE:</b> This bit is valid only for normal GPCM Mode.</p> <p>0 Abort mode. GPCM write access is terminated if IFCTA_B asserts before expiry of TWP counter in case of single beat transaction. If IFCTA_B is asserted during burst write the current transaction is aborted and only after deassertion of IFCTA_B it will be resumed. Details are given in <a href="#">Normal GPCM program operation, Figure 25-48</a>. Note that in abort mode no error is reported for write transaction.</p> <p>1 Acknowledgement mode: GPCM write access is acknowledged/qualified by external pin IFCTA_B assertion. If it is not asserted within CSOR[GPTO] time, GPCM_EVTER_STAT[TOER] will be set</p>
14–18 ADM_SHFT	<p>Address data multiplexing shift:</p> <p>Left shift the flash address by this value and assign it to address data multiplexed bus (AD[0:15]) ifc_data. By this method, the address msbs will be assigned to address data muxed bus that can be latched by asserting the AVD/ALE signal.</p> <p><b>NOTE:</b> This shifting is only valid in normal GPCM mode.</p> <p>Others Reserved</p> <p>00000 No shift</p>

*Table continues on the next page...*

**IFC\_CSORn\_GPCM field descriptions (continued)**

Field	Description
	00001 Left shift addr[0:31] by 1 and assign it to AD[0:15] 00010 Left shift ifc_addr by 2 00011 Left shift ifc_addr by 3 00100 Left shift ifc_addr by 4 00101 Left shift ifc_addr by 5 00110 Left shift ifc_addr by 6 00111 Left shift ifc_addr by 7 01000 Left shift ifc_addr by 8 01001 Left shift ifc_addr by 9 01010 Left shift ifc_addr by 10 01011 Left shift ifc_addr by 11 01100 Left shift ifc_addr by 12 01101 Left shift ifc_addr by 13 01110 Left shift ifc_addr by 14 01111 Left shift ifc_addr by 15 10000 Left shift ifc_addr by 16 10001 Left shift ifc_addr by 17 10010 Left shift ifc_addr by 18 10011 Left shift ifc_addr by 19 10100 Left shift ifc_addr by 20
19 -	This field is reserved.
20–22 BURST_LEN	GPCM burst length. It defines the maximum number of beats that will be sent/received in one burst cycle. This is programmed in terms of port-size transfer. For example, if the port size is 2 bytes and burst_length is 2, then the total of 8 bytes will be transferred in one burst cycle (that is, 4 beats of data transfers and each data transfer of 2 bytes). This is valid only in normal GPCM mode. 000 Non-burst mode 001 2 010 4 011 8 100 16 101 32 110 64 111 128
23–24 GAPERD	Generic ASIC parity error indication delay. Represents the delay (in terms of number of IFC_CLK) between the indication of parity error for address and write data with respect to start of frame (SOF_L). 00 1 IFC_CLK delayed 01 2 IFC_CLK delayed 10 3 IFC_CLK delayed 11 4 IFC_CLK delayed
25–26 -	This field is reserved.
27–29 TRHZ	Time for read enable high to output high impedance (Z)- Number of clocks required for memory to go in high Z after read enable deassertion.

*Table continues on the next page...*

**IFC\_CSORn\_GPCM field descriptions (continued)**

Field	Description
	If IFC is accessing for read operation, then after last byte read wait for these many clock cycles so that there is no contention on external bus.  Others Reserved  000 Wait for 20 IFC module input clocks 001 Wait for 40 IFC module input clocks 010 Wait for 60 IFC module input clocks 011 Wait for 80 IFC module input clocks 100 Wait for 100 IFC module input clocks
30 -	This field is reserved.
31 BCTL	Buffer control disable. This bit signifies presence or absence of external buffer. If buffer is absent then this bit should be set to 1.  0 BCTL is actively driven by IFC based on direction of access. 1 BCTL is set to its default value (high) irrespective of direction of access.

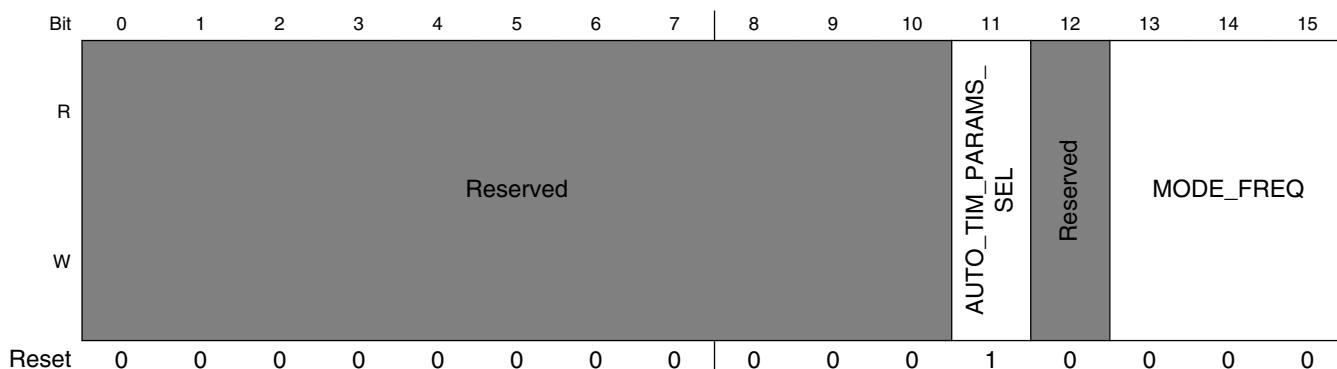
### 25.3.8 Extended Chip-Select Option register - NAND Flash Mode (IFC\_CSORn\_EXT)

The figure below shows the CSORn\_EXT fields when CSPRn[MSEL] selects the NAND flash mode.

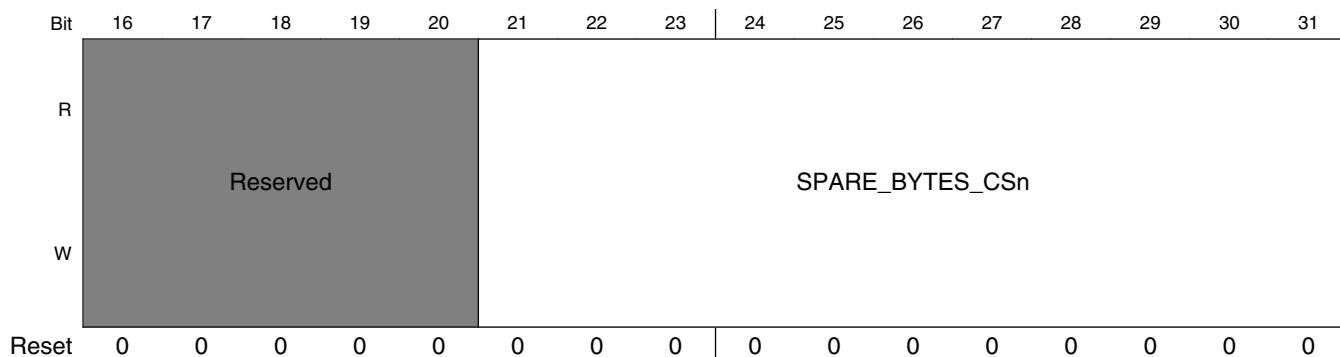
#### NOTE

CSORn\_EXT register is reserved if CSPRn[MSEL] is in NOR/GPCM mode.

Address: 153\_0000h base + 134h offset + (12d x i), where i=0d to 6d



## IFC memory map/register definition



### IFC\_CSOR<sub>n</sub>\_EXT field descriptions

Field	Description
0–10 -	This field is reserved.
11 AUTO_TIM_PARAMS_SEL	<p>Used to select the automatically calculated timing parameters instead of the user programmed timing parameters.</p> <p><b>NOTE:</b> Automatic parameter calculation option is not applicable for Async Mode. User has to compute and program FTIM registers for Async Mode.</p> <p>0 User programmed timing parameters (FTIM0-FTIM3) selected 1 Automatically calculated timing parameters selected</p>
12 -	This field is reserved.
13–15 MODE_FREQ	<p>Indicates the timing mode with which the source synchronous device is programmed. This field is qualified with NAND_MODE (CSOR[25:26]) and defined as follows:NAND_MODE = 01 (NVDDR mode)</p> <p>000 20 MHz 001 33 MHz 010 50 MHz 011 66 MHz 100 83 MHz 101 100 MHz 110 Reserved 111 Reserved</p>
16–20 -	This field is reserved.
21–31 SPARE_BYTES_CS <sub>n</sub>	<p>No. of bytes in spare region. This filed is applicable only when corresponding CSOR<sub>n</sub>[SPRZ] is 3'h110.</p> <p>Others Reserved.</p> <p>0x000 0 Spare region bytes 0x001 1 Spare region byte ... 0x400 1024 spare region bytes 0x7FE 2046 spare region bytes</p>

### 25.3.9 Flash Timing register 0 for Chip Select $n$ - NAND flash asyncNVDDR mode (IFC\_FTIM0\_CS $n$ \_NAND)

The flash timing registers define the flash interface timings. These register fields are defined differently depending on the machine type (NOR/NAND/GPCM) selected for that bank by CSPrn[MSEL] field. Timing registers are provided separately for each chip-select. Timing parameter are in terms of IFC module input clock cycles. More details about the register field is given in section [Programming model for flash interface timing](#).

#### NOTE

- Timing values of FTIM0, FTIM1, FTIM2, and FTIM3 registers for chip-select 0 in NAND/NOR flash mode will be loaded at boot\_load/rcw\_load by the parameters passed.
- If CSOR\_EXTn[AUTO\_TIM\_PARAMS\_SEL] is set, then the values returned on reading the FTIM0, FTIM1, FTIM2, FTIM3 registers will be the ones calculated automatically by IFC (based on CSOR\_EXTn[MODE\_FREQ] and the Clock division ratio) and not the ones programmed in the FTIM registers. This is valid only when CSORn[NAND\_MODE] = 2'b01, 2'b10, 2'b11.
- The TRP value loaded during RCW/boot load is taken as (TRAD +2) input clocks.
- For normal GPCM mode (write transaction) all the three timing parameters (that is, TEAHC, TACSE, and TCS) should not be programmed zero together.
- For normal GPCM mode (read transaction) all the three timing parameters (that is, TEAHC, TACSE, and TACO) should not be programmed zero together.

During NAND and NOR boot, default values of FTIM $n$ CS0 must be as shown in below table.

**Table 25-4. Default values of FTIM $n$ CS0 during boot**

FTIM $n$ CS0	Value during NAND boot	Value during NOR boot
FTIM0_CS0	181C_080C	C020_0410
FTIM1_CS0	3850_141A	5000_9028
FTIM2_CS0	0300_8028	0410_501C
FTIM3_CS0	2800_0000	0000_0000

## IFC memory map/register definition

The following figure shows the FTIM0\_CS $n$  fields when CSPRn[MSEL] selects the NAND flash mode and CSORn[NAND MODE] selects NAND flash async mode.

Address: 153\_0000h base + 1C0h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved				TCS				Reserved				TCAD			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W									Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### IFC\_FTIM0\_CS $n$ \_NAND field descriptions

Field	Description
0–1 -	This field is reserved.
2–7 TCS	Chip Enable (CE) setup time.  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks  ... 111111 63 IFC module input clocks
8–9 -	This field is reserved.
10–15 TCAD	Command, Address, Data delay (Command-command, command-address, address-address, address-command, command/address- start of data)  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks  ... 111111 63 IFC module input clocks
16–31 -	This field is reserved.

### 25.3.10 Flash Timing register 0 for Chip Select $n$ - NAND flash Asynchronous Mode (IFC\_FTIM0\_CS $n$ \_NAND\_ASYNC\_MODE)

The following register shows the FTIM0\_CS $n$  fields when CSPRn[MSEL] selects the NAND Flash Mode and CSORn[NAND\_MODE] selects NAND Flash Asynchronous Mode.

Address: 153\_0000h base + 1C0h offset + (48d  $\times$  i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved							Reserved								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IFC\_FTIM0\_CS $n$ \_NAND\_ASYNC\_MODE field descriptions

Field	Description
0 -	This field is reserved.
1–6 TCCST	This parameter is defined in the following two ways: Command latch enable (CLE) assertion time with respect to CS assertion time. CLE/ALE assertion time after previous CLE/ALE phase (post TWH time).  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks

Table continues on the next page...

**IFC\_FTIM0\_CS $n$ \_NAND\_ASYNC\_MODE field descriptions (continued)**

Field	Description
	...
	111111 63 IFC module input clocks
7 -	This field is reserved.
8–15 TWP	<p>Write enable (WE) pulse width</p> <p>00000000 Reserved</p> <p>00000001 1 IFC module input clock</p> <p>00000010 2 IFC module input clocks</p> <p>...</p> <p>11111111 255 IFC module input clocks</p>
16–17 -	This field is reserved.
18–23 TWCHT	<p>WE to command hold time</p> <p>000000 Reserved</p> <p>000001 1 IFC module input clock</p> <p>000010 2 IFC module input clocks</p> <p>...</p> <p>111111 63 IFC module input clocks</p>
24–25 -	This field is reserved.
26–31 TWH	<p>WE high hold time</p> <p>000000 Reserved</p> <p>000001 1 IFC module input clock</p> <p>000010 2 IFC module input clocks</p> <p>...</p> <p>111111 63 IFC module input clocks</p>

### 25.3.11 Flash Timing register 0 for CSn - NOR Flash Mode (IFC\_FTIM0\_CSn\_NOR)

The following figure shows the FTIM0\_CS<sub>n</sub> fields when CSPR<sub>n</sub>[MSEL] selects the NOR flash mode.

Address: 153\_0000h base + 1C0h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	TACSE				Reserved				TEADC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved		TAVDS				Reserved		TEAHC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_FTIM0\_CS<sub>n</sub>\_NOR field descriptions**

Field	Description
0–3 TACSE	Address phase (after external latch enable deassertion) end to chip enable assertion time 0000 Reserved 0001 1 IFC module input clock 0010 2 IFC module input clocks ... 1111 15 IFC module input clocks
4–9 -	This field is reserved.
10–15 TEADC	External latch address delay cycles: External address valid (AVD) assertion time (pulse width of external latch enable signal). 000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks ... 111111 63 IFC module input clocks
16–17 -	This field is reserved.
18–23 TAVDS	Delay between CS assertion to AVD/ALE assertionin AVD devices (address latch internal to the device) 000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks ... 111111 63 IFC module input clocks

Table continues on the next page...

**IFC\_FTIM0\_CS<sub>n</sub>\_NOR field descriptions (continued)**

Field	Description										
24–25 -	This field is reserved.										
26–31 TEAHC	<p>Latch address hold cycles.</p> <table> <tr><td>000000</td><td>Reserved</td></tr> <tr><td>000001</td><td>1 IFC module input clock</td></tr> <tr><td>000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>111111</td><td>63 IFC module input clocks</td></tr> </table>	000000	Reserved	000001	1 IFC module input clock	000010	2 IFC module input clocks	...		111111	63 IFC module input clocks
000000	Reserved										
000001	1 IFC module input clock										
000010	2 IFC module input clocks										
...											
111111	63 IFC module input clocks										

### 25.3.12 Flash Timing register 0 for CS<sub>n</sub> - Normal GPCM Mode (IFC\_FTIM0\_CS<sub>n</sub>\_GPCM)

The following figure shows the FTIM0\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the GPCM mode.

Address: 153\_0000h base + 1C0h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TACSE	Reserved				TEADC				Reserved				TEAHC																		
W	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**IFC\_FTIM0\_CS<sub>n</sub>\_GPCM field descriptions**

Field	Description										
0–3 TACSE	<p>Address phase (After address hold cycle) end to chip enable assertion time</p> <table> <tr><td>0000</td><td>Reserved</td></tr> <tr><td>0001</td><td>1 IFC module input clock</td></tr> <tr><td>0010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>1111</td><td>15 IFC module input clocks</td></tr> </table>	0000	Reserved	0001	1 IFC module input clock	0010	2 IFC module input clocks	...		1111	15 IFC module input clocks
0000	Reserved										
0001	1 IFC module input clock										
0010	2 IFC module input clocks										
...											
1111	15 IFC module input clocks										
4–9 -	This field is reserved.										
10–15 TEADC	<p>External latch address delay cycles. External address latch enable (ALE) assertion time (pulse width of external latch enable signal)</p> <table> <tr><td>000000</td><td>Reserved</td></tr> <tr><td>000001</td><td>1 IFC module input clock</td></tr> <tr><td>000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>111111</td><td>63 IFC module input clocks</td></tr> </table>	000000	Reserved	000001	1 IFC module input clock	000010	2 IFC module input clocks	...		111111	63 IFC module input clocks
000000	Reserved										
000001	1 IFC module input clock										
000010	2 IFC module input clocks										
...											
111111	63 IFC module input clocks										

Table continues on the next page...

**IFC\_FTIM0\_CS<sub>n</sub>\_GPCM field descriptions (continued)**

Field	Description
16–25 -	This field is reserved.
26–31 TEAHC	External latch address hold cycles: Address hold cycle relative to external latch enable signal  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks ... 111111 63 IFC module input clocks

**25.3.13 Flash Timing register 1 for Chip-Select n - NAND Flash NVDDR Mode (IFC\_FTIM1\_CS<sub>n</sub>\_NAND)**

The following figure shows the FTIM1\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NAND flash mode.

Address: 153\_0000h base + 1C4h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	TADLE							TWB								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved		TRR						TWRCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_FTIM1\_CS<sub>n</sub>\_NAND field descriptions**

Field	Description
0–7 TADLE	Effective address to data loading time  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clocks
8–15 TWB	Clock Rising Edge to SR[6] (R/B) low
16–17 -	This field is reserved.
18–23 TRR	Ready busy high to read enable (RE) low time  000000 Reserved

*Table continues on the next page...*

**IFC\_FTIM1\_CS<sub>n</sub>\_NAND field descriptions (continued)**

Field	Description
	000001 1 IFC module input clock 000010 2 IFC module input clocks ... 111111 63 IFC module input clocks
24–31 TWRCK	W/R low to data output cycle

**25.3.14 Flash Timing register 1 for Chip Select n - Asynchronous Mode (IFC\_FTIM1\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE)**

The following figure shows the FTIM1\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NAND flash asynchronous mode.

Address: 153\_0000h base + 1C4h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_FTIM1\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE field descriptions**

Field	Description
0–7 TADLE	Effective address to data loading time 00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clocks
8–15 TWBE	WE high (after TWH time) to ready busy (RB_B) low time 00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11101111 239 IFC module input clocks 11110000-11111111 Reserved

Table continues on the next page...

**IFC\_FTIM1\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE field descriptions (continued)**

Field	Description										
16–17 -	This field is reserved.										
18–23 TRR	<p>Ready busy high to read enable (RE) low time</p> <table> <tr><td>000000</td><td>Reserved</td></tr> <tr><td>000001</td><td>1 IFC module input clock</td></tr> <tr><td>000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>111111</td><td>63 IFC module input clocks</td></tr> </table>	000000	Reserved	000001	1 IFC module input clock	000010	2 IFC module input clocks	...		111111	63 IFC module input clocks
000000	Reserved										
000001	1 IFC module input clock										
000010	2 IFC module input clocks										
...											
111111	63 IFC module input clocks										
24–31 TRP	<p>RE pulse width</p> <table> <tr><td>00000000</td><td>Reserved</td></tr> <tr><td>00000001</td><td>1 IFC module input clock</td></tr> <tr><td>00000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>11111111</td><td>255 IFC module input clocks</td></tr> </table>	00000000	Reserved	00000001	1 IFC module input clock	00000010	2 IFC module input clocks	...		11111111	255 IFC module input clocks
00000000	Reserved										
00000001	1 IFC module input clock										
00000010	2 IFC module input clocks										
...											
11111111	255 IFC module input clocks										

### 25.3.15 Flash Timing register 1 for CS<sub>n</sub> - NOR Flash Mode (IFC\_FTIM1\_CS<sub>n</sub>\_NOR)

The following figure shows the FTIM1\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NOR flash mode.

Address: 153\_0000h base + 1C4h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TACO								Reserved								TRAD_NOR								TSEQRAD_NOR							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_FTIM1\_CS<sub>n</sub>\_NOR field descriptions**

Field	Description										
0–7 TACO	<p>This field represents CS_B assertion to output enable (OE) assertion setup time for conventional NOR and address hold time to OE_B assertion in case of ADM NOR</p> <table> <tr><td>00000000</td><td>Reserved</td></tr> <tr><td>00000001</td><td>1 IFC module input clock</td></tr> <tr><td>00000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>11111111</td><td>255 IFC module input clocks</td></tr> </table>	00000000	Reserved	00000001	1 IFC module input clock	00000010	2 IFC module input clocks	...		11111111	255 IFC module input clocks
00000000	Reserved										
00000001	1 IFC module input clock										
00000010	2 IFC module input clocks										
...											
11111111	255 IFC module input clocks										
8–15 -	This field is reserved.										

Table continues on the next page...

**IFC\_FTIM1\_CS<sub>n</sub>\_NOR field descriptions (continued)**

Field	Description
16–23 TRAD_NOR	NOR flash read access delay: It represents the read enable to data access time plus total round trip board delay from the external NOR flash memory during read operation. Its value can be calculated as <i>Read data access time considering data gets sampled in the mid of data eye + 2* Board Delay + 2</i> .  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clocks
24–31 TSEQRAD_NOR	NOR flash sequential read access delay. It represents the address to data access time plus total round trip delay from the external NOR flash memory during sequential read operation. Its value can be calculated as [ <i>NOR Page Address Delay (max) for sequential read + 2*Board Delay + Device Setup Time</i> ].  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ..... 11111111 255 IFC module input clocks

### 25.3.16 Flash Timing register 1 for CS<sub>n</sub> - Normal GPCM Mode (IFC\_FTIM1\_CS<sub>n</sub>\_GPCM)

The following figure shows the FTIM1\_CS<sub>n</sub> fields when CSPRn[MSEL] and CSOPn[GPMODE] selects the normal GPCM mode.

Address: 153\_0000h base + 1C4h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_FTIM1\_CS<sub>n</sub>\_GPCM field descriptions**

Field	Description
0–7 TACO	CS assertion to output enable (OE) assertion setup time  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clock
8–17 -	This field is reserved.

Table continues on the next page...

**IFC\_FTIM1\_CS<sub>n</sub>\_GPCM field descriptions (continued)**

Field	Description
18–23 TRAD	GPCM read access delay: It represents the output enable assertion time. See <a href="#">Normal GPCM read operation</a> .  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks ... 111111 63 IFC module input clocks
24–31 -	This field is reserved.

**25.3.17 Flash Timing register 2 for Chip Select n - NAND Flash NVDDR Mode (IFC\_FTIM2\_CS<sub>n</sub>\_NAND)**

The following figure shows the FTIM2\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NAND flash mode.

Address: 153\_0000h base + 1C8h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TCKWR							TWHR							TRHW							Reserved										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_FTIM2\_CS<sub>n</sub>\_NAND field descriptions**

Field	Description
0–7 TCKWR	Data output end to W/R high.
8–15 TWHR	Command, Address or Data input cycle to Data output cycle.
16–23 TRHW	Data Output cycle to Command, Address or Data Input cycle
24–31 -	This field is reserved.

**25.3.18 Flash Timing register 2 for Chip Select n - NAND Flash Asynchronous Mode (IFC\_FTIM2\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE)**

The following figure shows the FTIM2\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NAND flash asynchronous mode.

## IFC memory map/register definition

Address: 153\_0000h base + 1C8h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		TRAD						Reserved				TREH				Reserved				TWHRE											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## IFC\_FTIM2\_CSn\_NAND\_ASYNC\_MODE field descriptions

Field	Description
0–2 -	This field is reserved.
3–10 TRAD	Flash read access delay: It represents the data sampling time of read data. It should be programmed such that sampling is done at the center of the received data eye. For calculation of data eye width and appropriate data sampling time refer to <a href="#">NAND asynchronous mode calculating read data window width</a> .  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks  ... 11111111 255 IFC module input clocks
11–14 -	This field is reserved.
15–20 TREH	RE_B High Time  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks  ... 111111 63 IFC module input clocks
21–23 -	This field is reserved.
24–31 TWHRE	WE_B High to RE_B Low Effective  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks  ... 11111111 255 IFC module input clocks

### 25.3.19 Flash Timing register 2 for CSn - NOR Flash Mode (IFC\_FTIM2\_CSn\_NOR)

The following figure shows the FTIM2\_CS<sub>n</sub> fields when CSPR<sub>n</sub>[MSEL] selects the NOR flash mode.

Address: 153\_0000h base + 1C8h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved				TCS				Reserved		TCH				Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	TWPH						Reserved		TWP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_FTIM2\_CS<sub>n</sub>\_NOR field descriptions

Field	Description
0–3 -	This field is reserved.
4–7 TCS	Chip-select assertion to WE assertion setup time  0000 Reserved 0001 1 IFC module input clock 0010 2 IFC module input clocks  ... 1111 15 IFC module input clocks
8–9 -	This field is reserved.
10–13 TCH	Chip-select hold time with respect to WE deassertion  0000 Reserved 0001 1 IFC module input clock 0010 2 IFC module input clocks  ... 1111 15 IFC module input clocks
14–15 -	This field is reserved.
16–21 TWPH	Write enable pulse high time  000000 Reserved 000001 1 IFC module input clock 000010 2 IFC module input clocks  ... 111111 63 IFC module input clocks

Table continues on the next page...

**IFC\_FTIM2\_CS<sub>n</sub>\_NOR field descriptions (continued)**

Field	Description
22–23 -	This field is reserved.
24–31 TWP	Write enable pulse width  00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clocks

### 25.3.20 Flash Timing register 2 for CS<sub>n</sub> - Normal GPCM Mode (IFC\_FTIM2\_CS<sub>n</sub>\_GPCM)

The following figure shows the FTIM2\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the normal GPCM mode.

Address: 153\_0000h base + 1C8h offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved				TCS				Reserved		TCH				Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved								TWP							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_FTIM2\_CS<sub>n</sub>\_GPCM field descriptions**

Field	Description
0–3 -	This field is reserved.
4–7 TCS	Chip-select assertion to WE assertion setup time  0000 Reserved 0001 1 IFC module input clock 0010 2 IFC module input clocks ... 1111 15 IFC module input clocks
8–9 -	This field is reserved.
10–13 TCH	Chip-select hold time with respect to WE deassertion  0000 Reserved

*Table continues on the next page...*

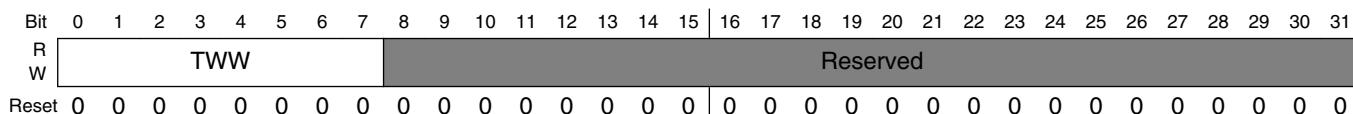
**IFC\_FTIM2\_CS*n*\_GPCM field descriptions (continued)**

Field	Description
	0001 1 IFC module input clock 0010 2 IFC module input clocks ... 1111 15 IFC module input clocks
14–23 -	This field is reserved.
24–31 TWP	Write enable pulse width 00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clock

### 25.3.21 Flash Timing register 3 for Chip Select *n* - NAND Flash Mode (IFC\_FTIM3\_CS*n*\_NAND)

The following figure shows the FTIM3\_CS*n* fields when CSPR*n*[MSEL] selects the NAND flash NVDDRmode.

Address: 153\_0000h base + 1CCh offset + (48d × i), where i=0d to 6d

**IFC\_FTIM3\_CS*n*\_NAND field descriptions**

Field	Description
0–7 TWW	Write Protect WP_B transition time. It represents the idle cycles inserted by the IFC controller whenever corresponding WP_B pin toggles. Refer <a href="#">Write protect</a> for more details. 00000000 Reserved 00000001 1 IFC module input clock 00000010 2 IFC module input clocks ... 11111111 255 IFC module input clocks
8–31 -	This field is reserved.

### 25.3.22 Flash Timing register 3 for Chip Select n - NAND Flash Asynchronous Mode (IFC\_FTIM3\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE)

The following figure shows the FTIM3\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NAND flash asynchronous mode.

Address: 153\_0000h base + 1CCh offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TWW								Reserved																							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### IFC\_FTIM3\_CS<sub>n</sub>\_NAND\_ASYNC\_MODE field descriptions

Field	Description										
0–7 TWW	<p>Write Protect WP_B transition time. It represents the idle cycles inserted by the IFC controller whenever corresponding WP_B pin toggles.</p> <p>Refer <a href="#">Write protect</a> for more details.</p> <table> <tr><td>00000000</td><td>Reserved</td></tr> <tr><td>00000001</td><td>1 IFC module input clock</td></tr> <tr><td>00000010</td><td>2 IFC module input clocks</td></tr> <tr><td>...</td><td></td></tr> <tr><td>11111111</td><td>255 IFC module input clocks</td></tr> </table>	00000000	Reserved	00000001	1 IFC module input clock	00000010	2 IFC module input clocks	...		11111111	255 IFC module input clocks
00000000	Reserved										
00000001	1 IFC module input clock										
00000010	2 IFC module input clocks										
...											
11111111	255 IFC module input clocks										
8–31 -	This field is reserved.										

### 25.3.23 Flash Timing register 3 for CS<sub>n</sub> - NOR Flash Mode (IFC\_FTIM3\_CS\_NOR)

The following figure shows the FTIM3\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the NOR flash mode.

#### NOTE

Offset for IFC\_FTIM3\_CS<sub>n</sub>\_NOR are 0x0\_01CC, 0x0\_01FC, 0x0\_022C, 0x0\_025C, 0x0\_028C, 0x0\_02BC, and 0x0\_02EC

Address: 153\_0000h base + 1CCh offset = 153\_01CCh

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved																															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**IFC\_FTIM3\_CS\_NOR field descriptions**

Field	Description
0–31 -	This field is reserved.

**25.3.24 Flash Timing register 3 for CSn - Normal GPCM Mode (IFC\_FTIM3\_CS<sub>n</sub>\_GPCM)**

The following figure shows the FTIM3\_CS<sub>n</sub> fields when CSPRn[MSEL] selects the GPCM flash mode.

Address: 153\_0000h base + 1CCh offset + (48d × i), where i=0d to 6d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TAAD					Reserved																										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_FTIM3\_CS<sub>n</sub>\_GPCM field descriptions**

Field	Description										
0–5 TAAD	<p>GPCM address access delay. This timing parameter is required for read cycles when GPCM is working in burst mode, that is, burst length is programmed to a non-zero value. The first data beat of a burst is sampled after TRAD time of OE assertion; subsequent beat data are sampled after TAAD time of the previous sample pulse. The same timing is used for sending the new address of the beat. The second address is sent after TRAD time of OE assertion and subsequent address beats are sent after TAAD time of the last beat. This is programmed in terms of IFC module input clock but should always be a multiple of IFC_CLK.</p> <p>This is valid only for normal GPCM mode.</p> <table> <tr> <td>000000</td> <td>Reserved</td> </tr> <tr> <td>000001</td> <td>1 IFC module input clock</td> </tr> <tr> <td>000010</td> <td>2 IFC module input clocks</td> </tr> <tr> <td>...</td> <td></td> </tr> <tr> <td>111111</td> <td>63 IFC module input clocks</td> </tr> </table>	000000	Reserved	000001	1 IFC module input clock	000010	2 IFC module input clocks	...		111111	63 IFC module input clocks
000000	Reserved										
000001	1 IFC module input clock										
000010	2 IFC module input clocks										
...											
111111	63 IFC module input clocks										
6–31 -	This field is reserved.										

### 25.3.25 Ready Busy Status for each Chip Select (IFC\_RB\_STAT)

Address: 153\_0000h base + 400h offset = 153\_0400h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RB0	RB1	RB2	RB3	RB4	RB5	RB6		Reserved							
W																
Reset	n	n	n	n	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_RB\_STAT field descriptions

Field	Description
0 RB0	<p>Ready Busy input from CS0. This register field represents the status of RB_B signal of CS0 sampled by IFC at every clock.</p> <p>0 Device connected at CS0 is driving RB_B signal as Busy 1 Device connected at CS0 is driving RB_B signal as Ready.</p>
1 RB1	<p>Ready Busy input from CS1</p> <p>This register field represents the status of RB_B signal of CS1 sampled by IFC at every clock.</p> <p>0 Device connected at CS1 is driving RB_B signal as Busy 1 Device connected at CS1 is driving RB_B signal as Ready.</p>
2 RB2	<p>Ready Busy input from CS2</p> <p>This register field represents the status of RB_B signal of CS2 sampled by IFC at every clock.</p> <p>0 Device connected at CS2 is driving RB_B signal as Busy 1 Device connected at CS2 is driving RB_B signal as Ready.</p>
3 RB3	<p>Ready Busy input from CS3</p> <p>This register field represents the status of RB_B signal of CS3 sampled by IFC at every clock.</p> <p>0 Device connected at CS3 is driving RB_B signal as Busy 1 Device connected at CS3 is driving RB_B signal as Ready.</p>
4 RB4	<p>Ready Busy input from CS4</p> <p>This register field represents the status of RB_B signal of CS4 sampled by IFC at every clock.</p> <p>0 Device connected at CS4 is driving RB_B signal as Busy 1 Device connected at CS4 is driving RB_B signal as Ready.</p>
5 RB5	<p>Ready Busy input from CS5</p> <p>This register field represents the status of RB_B signal of CS5 sampled by IFC at every clock.</p> <p>0 Device connected at CS5 is driving RB_B signal as Busy 1 Device connected at CS5 is driving RB_B signal as Ready.</p>

Table continues on the next page...

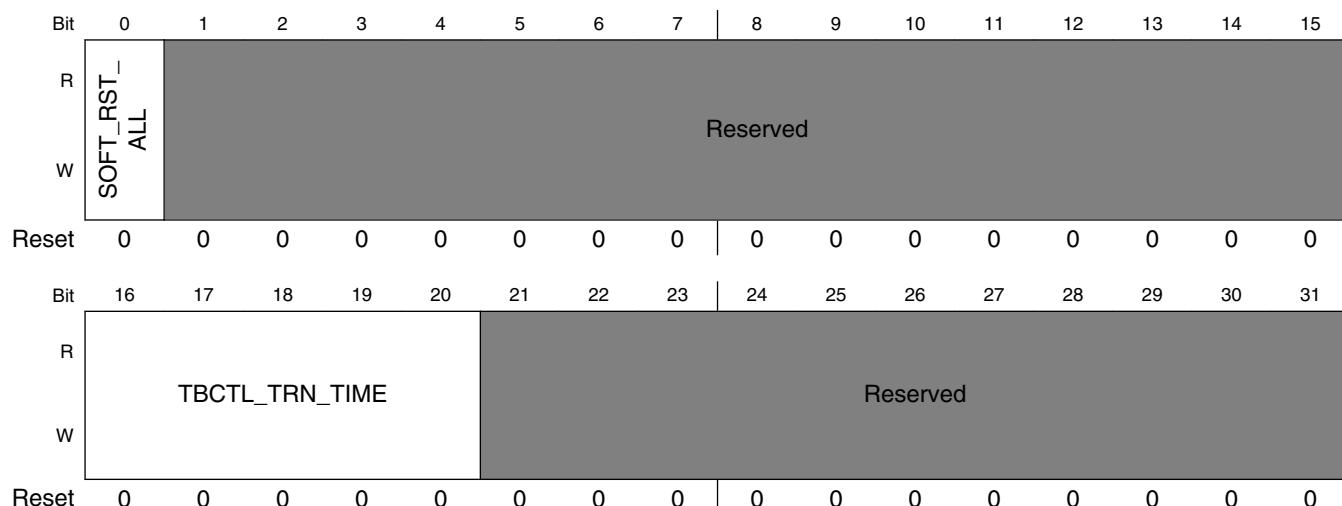
**IFC\_RB\_STAT field descriptions (continued)**

Field	Description
6 RB6	Ready Busy input from CS6  This register field represents the status of RB_B signal of CS6 sampled by IFC at every clock.  0 Device connected at CS6 is driving RB_B signal as Busy 1 Device connected at CS6 is driving RB_B signal as Ready.
7–31 -	This field is reserved.

**25.3.26 General Control register (IFC\_GCR)**

The reset value of this field is passed through parameter.

Address: 153\_0000h base + 40Ch offset = 153\_040Ch

**IFC\_GCR field descriptions**

Field	Description
0 SOFT_RST_ALL	Software Reset All: It is used to reset the whole IFC hardware (NAND, NOR, and GPCM).  With this reset only the event/error status registers of IFC will be cleared. Programming registers will retain their value.  The software reset mechanism is implemented such that IFC will send the pending transactions on system bus with good response. In case of pending read transaction, garbage data will be supplied back. Once the whole IFC is in IDLE state, the software can clear this bit. IFC hardware stops software from clearing this bit till all the processing has been done and IFC hardware is in idle state.  If this bit is set, hardware will clear it once reset operation is completed. Software should poll this bit to get cleared before initiating any new transaction.  0 No Software reset 1 Assert Software reset

Table continues on the next page...

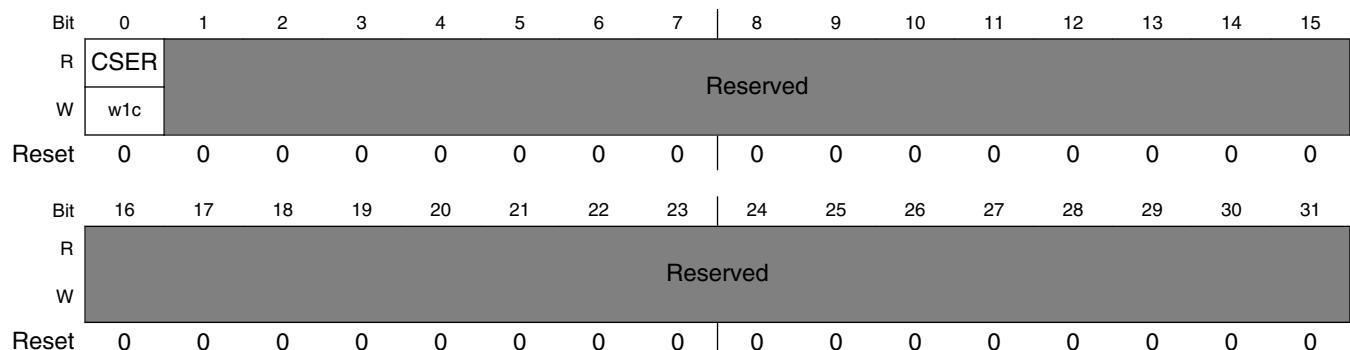
**IFC\_GCR field descriptions (continued)**

Field	Description
1–15 -	This field is reserved.
16–20 TBCTL_TRN_TIME	It represents the turnaround time of external buffer in terms of number of IFC module input clock cycles. BCTL should be kept high for these many clock cycles before issuing a write transaction after a read on flash interface. See <a href="#">Data buffer control (BCTL)</a> for more details.
21–31 -	This field is reserved.

### 25.3.27 Common Event and Error Status register (IFC\_CM\_EVTER\_STAT)

Common event and error status register (CM\_EVTER\_STAT) indicates the cause of an error or event that cannot be classified in NAND, NOR, and GPCM.

Address: 153\_0000h base + 418h offset = 153\_0418h

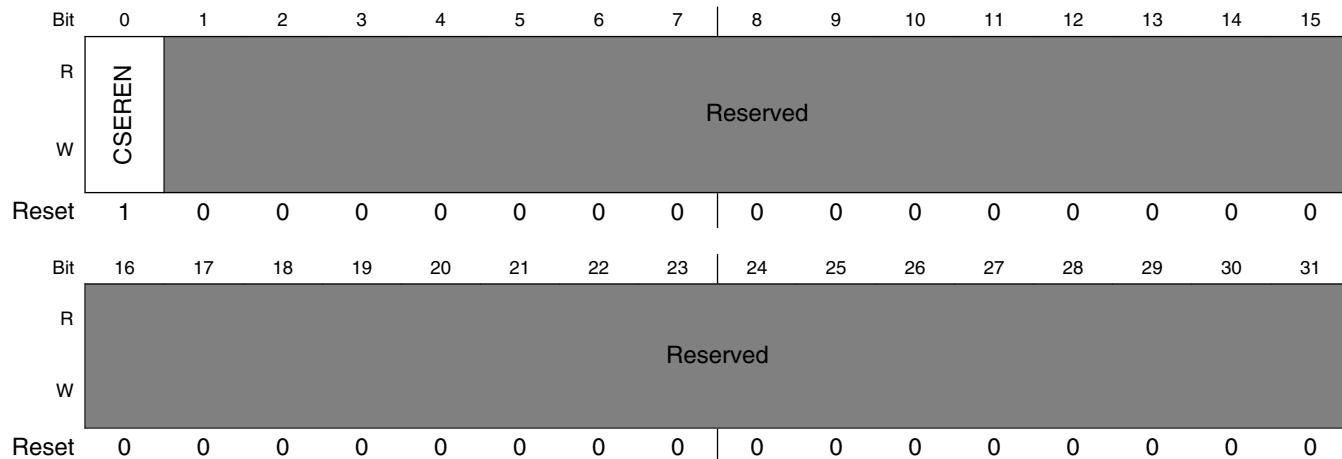
**IFC\_CM\_EVTER\_STAT field descriptions**

Field	Description
0 CSER	Chip-Select Error 0 No chip-select error 1 A transaction was sent to IFC which is not mapped to any memory bank
1–31 -	This field is reserved.

### 25.3.28 Common Event and Error Enable register (IFC\_CM\_EVTER\_EN)

This register is used to enable/disable the logging of event and error indication in the CM\_EVTER\_STAT register.

Address: 153\_0000h base + 424h offset = 153\_0424h



#### IFC\_CM\_EVTER\_EN field descriptions

Field	Description
0 CSEREN	Chip-Select Error Checking Enable 0 Chip-Select Error Checking Disabled 1 Chip-Select Error Checking Enabled
1–31 -	This field is reserved.

### 25.3.29 Common Event and Error Interrupt Enable register (IFC\_CM\_EVTER\_INTR\_EN)

Common event and error interrupt enable register (CM\_EVTER\_INTR\_EN) is used to enable/disable the generation of interrupt signals corresponding to common error and event reporting. Software must clear pending events and errors in CM\_EVTER\_STAT register before enabling the interrupts.

Address: 153\_0000h base + 430h offset = 153\_0430h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CSERIREN															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_CM\_EVTER\_INTR\_EN field descriptions

Field	Description
0 CSERIREN	Chip-Select Error Interrupt Enable 0 Chip-Select Error Interrupt Disabled 1 Chip-Select Error Interrupt Enabled
1–31 -	This field is reserved.

### 25.3.30 Common Transfer Error Attributes register 0 (IFC\_CM\_ERATTR0)

These registers store the attribute corresponding to the first transaction on which common error occurred.

Common transfer error attribute register 0 (CM\_ERATTR0) is used to register the transaction attributes corresponding to the first common error occurred.

Address: 153\_0000h base + 43Ch offset = 153\_043Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ERTYP	Reserved			ERAID						Reserved					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ERSRCID								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_CM\_ERATTR0 field descriptions**

Field	Description
0 ERTYP	Transaction type of the error 0 Write Transaction 1 Read Transaction
1–3 -	This field is reserved.
4–11 ERAID	ID of the error transaction
12–15 -	This field is reserved.
16–23 ERSRCID	Source ID of the error transaction, always zero.
24–31 -	This field is reserved.

### 25.3.31 Common Transfer Error Attributes register 1 (IFC\_CM\_ERATTR1)

Common transfer error attribute register1 (CM\_ERATTR1) is used to register the transaction address corresponding to the first error.

Address: 153\_0000h base + 440h offset = 153\_0440h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ERADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### IFC\_CM\_ERATTR1 field descriptions

Field	Description																												
0–31 ERADDR	32 bits of transaction address corresponding to error transaction																												

### 25.3.32 Clock Control register (IFC\_CCR)

Address: 153\_0000h base + 44Ch offset = 153\_044Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31													
R	Reserved																																												
W																																													
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	R	Reserved																											
W																																													
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															

#### IFC\_CCR field descriptions

Field	Description																												
0–3 -	This field is reserved.																												

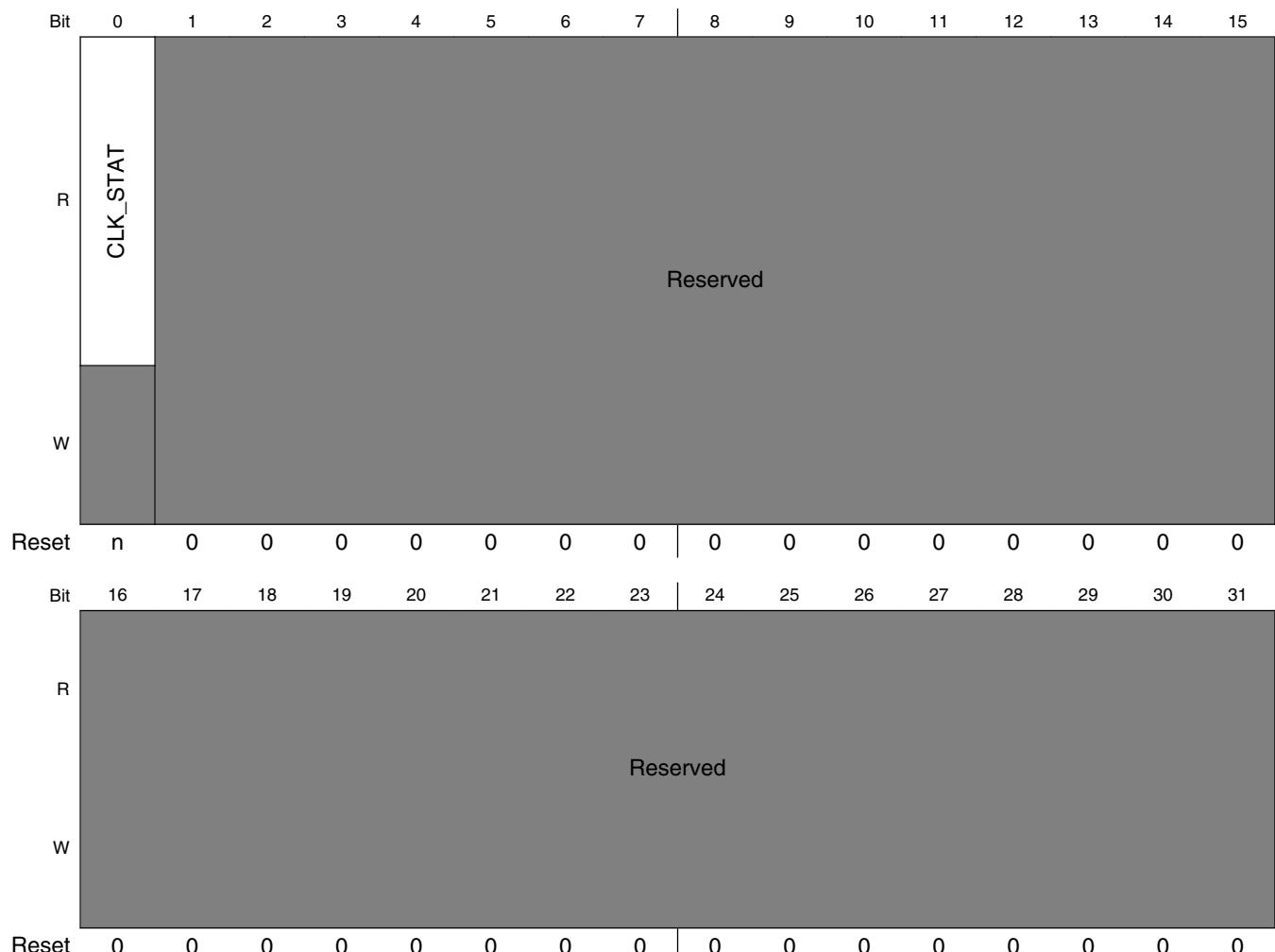
Table continues on the next page...

**IFC\_CCR field descriptions (continued)**

Field	Description
4–7 CLKDIV	Clock division ratio: This field is used to generate the IFC clock from the IFC module input clock. The IFC clock is sent out after inverting it according to the INV_CLK_EN setting.  0000 Reserved 0001 Divide by 2 0010 Divide by 3 0011 Divide by 4 0100 Reserved 0101 Divide by 6 0110 Reserved 0111 Divide by 8 1000 Reserved 1001 Reserved 1010 Reserved 1011 Divide by 12 1100 Reserved 1101 Reserved 1110 Reserved 1111 Divide by 16
8–11 -	This field is reserved.
12–15 CLK_DLY	IFC Clock Delay  This field specifies the number of IFC module input clocks by which external clock is delayed.  0000 No delay 0001 1 IFC module input clocks delay introduced 0010 2 IFC module input clocks delay introduced ... 1111 15 IFC module input clocks delay introduced
16 INV_CLK_EN	IFC Clock Inversion  This field specifies whether IFC clock is inverted before sending out.  0 IFC clock is not inverted 1 IFC clock is inverted
17–31 -	This field is reserved.

### 25.3.33 Clock Status register (IFC\_CSR)

Address: 153\_0000h base + 450h offset = 153\_0450h



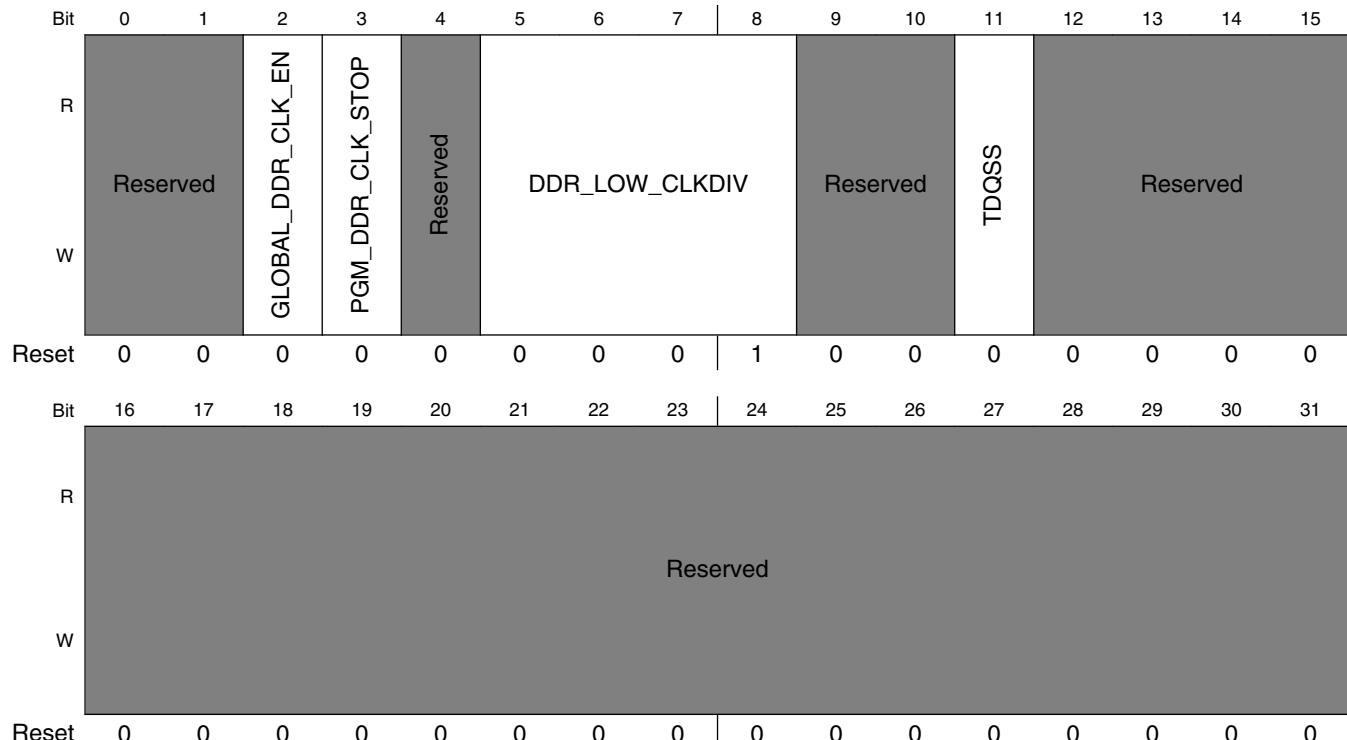
#### IFC\_CSR field descriptions

Field	Description
0 CLK_STAT	Clock Status. This bit is cleared by hardware in case any of the field of CCR is changed. It is again set when external clock is stable. New GPCM operation can be initiated only after clock is stable for new CLK_DIV ratio.  0 Clock is unstable 1 Clock is stable
1–31 -	This field is reserved.

### 25.3.34 DDR Clock Control register (IFC\_DDR\_CCR)

This register is used for programming the clock divider when NV-DDR interface is at frequencies up to 100 MHz.

Address: 153\_0000h base + 454h offset = 153\_0454h



**IFC\_DDR\_CCR field descriptions**

Field	Description
0–1 -	This field is reserved. Reserved
2 GLOBAL_DDR_CLK_EN	Switches off the IFC DDR clock. Used when device is not programmed as Source Synchronous. 0 Clock disabled. 1 Clock enabled.
3 PGM_DDR_CLK_STOP	DDR Clock Stop in case of Program Operation (except SET FEATURES). This field is valid only when CSOR[NAND_MODE] = 2'b01 0 Clock Enabled 1 Clock Stopped
4 -	This field is reserved. Reserved

*Table continues on the next page...*

**IFC\_DDR\_CCR field descriptions (continued)**

Field	Description
5–8 DDR_LOW_CLKDIV	Clock division ratio: This field is used to generate the divided clock from ip_clk.  0000- Divide by 2 0001- Divide by 4 0010- Divide by 6 0011- Divide by 8 0100- Divide by 10 0101- Divide by 12 0110- Divide by 14 0111- Divide by 16 1000- Divide by 18 1001- Divide by 20 1010- Divide by 22 1011- Divide by 24 1100- Divide by 26 1101- Divide by 28 1110- Divide by 30 1111- Divide by 32  Note- After reset, divided by 4 clock is sent out
9–10 -	This field is reserved. Reserved
11 TDQSS	Data input to first DQS transition delay. This field is valid only when CSOR[NAND_MODE] = 2'b01 0 0.75 IFC_DDR_CLOCK period 1 1.25 IFC_DDR_CLOCK period
12–31 -	This field is reserved.

### 25.3.35 NAND Configuration register (IFC\_NCFGR)

A separate 1 KB memory-mapped region is assigned to NAND-specific registers.

Address: 153\_0000h base + 1000h offset = 153\_1000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	BOOT	Reserved	SRAM_INIT_EN	Reserved	SINGLE_DATA_MODE	Reserved	Reserved	Reserved	ADDR_MODE	Reserved						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	NUM_LOOP	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	NUM_WAIT	Reserved						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_NCFGR field descriptions

Field	Description
0 BOOT	NAND flash auto-boot load mode: During system boot from NAND flash, this bit remains set to alter the use of the FCM buffer RAM. Software should clear BOOT once FCM is to be restored for normal operation. Setting BOOT without auto-boot in progress only alters the mapping of the buffer RAM.  0 NAND FCM is operating in normal functional mode, with a 16-KB FCM buffer RAM. In this case normal SRAM buffer mapping applies. 1 Flash is accessed in boot mode. SRAM buffer mapping gets changed and whole 8 KB NAND flash main area in the buffer looks as contiguous linear addressable memory region. This bit is set by hardware when por_cfg_rcw_load or por_cfg_boot_load occurs with NAND as bootable device.
1 -	This field is reserved.
2 SRAM_INIT_EN	SRAM Initialization Enable This bit is provided to trigger the auto initialization of complete SRAM with FF data. Software must set this bit before initiating first program operation on NAND post reset. IFC clears this bit after finishing this operation.

Table continues on the next page...

## IFC\_NCFGR field descriptions (continued)

Field	Description
	<p>After setting this bit to 1, software must poll this bit. Reading back value 0 confirms that IFC hardware has filled the SRAM and cleared this bit.</p> <p>0 No auto SRAM Initialization. 1 Triggers the IFC sram initialization logic which fills all sram buffer locations with 'hFF data</p>
3 -	This field is reserved.
4 SINGLE_DATA_MODE	<p>This bit is valid only when CSORn[NAND_MODE] is configured as NV-DDR Mode.</p> <p>In case of source synchronous operation the NAND flash repeats a byte of data on both the rising and falling edge of the DQS for certain commands, namely GET FEATURES, READ ID. For the SET FEATURES command, IFC has to repeat each byte of data for the rising and falling edge of DQS.</p> <p>When this bit is set:</p> <p>For read operations (GET FEATURES, READ ID), IFC will ignore the repeated bytes of data sent by the NAND Flash on the falling edge of the DQS for read operations.</p> <p>For write operations (SET FEATURES), IFC will replicate each byte of data to be written on both rising and falling edge of DQS.</p> <p>0 Single data mode disabled 1 Single data mode enabled</p>
5–7 -	This field is reserved.
8–9 ADDR_MODE	<p>Addressing Mode: Applicable for the current active CS. Applicable only with opcodes CA0 and RA0.</p> <p>Others Reserved</p> <p>00 ROW0/COL0, ROW0+1/COL0, ROW0+2/COL0 and so on, in each iteration defined by NUM_LOOP field (Incremental by Page Size) 01 ROW0/COL0 addresses in first iteration, ROW1/COL1 address in second iteration, ROW2/COL2 address in third iteration, ROW3/COL3 address in fourth iteration</p>
10–15 -	This field is reserved.
16–19 NUM_LOOP	<p>Number of loop iterations of FIR sequences for multipage operations</p> <p>FIR sequence will execute lopping by NUM_LOOP's time.</p> <p><b>NOTE:</b> 1 NUM_LOOP value should be programmed carefully as it is restricted by SRAM buffer size. We support 16-page operation for small page, 4-page operation for large page of size 2 KB and 2-page operations for 4 KB page size. Hence NUM_LOOP value of 0000-1111 is only valid for small page; for 2 KB page size NUM_LOOP 0000-0011 are valid values; and for 4-KB page size NUM_LOOP value 0000 and 0001 are valid.</p> <p><b>NOTE:</b> 2 For ADDR_MODE = 01, NUM_LOOP should not exceed 4.</p> <p><b>NOTE:</b> 3 Ensure that separate SRAM buffers should be allocated during consecutive runs of NUM_LOOP else the data from the previous iteration will be overwritten and produce incorrect results.</p> <p>0000 1 time execution of FIR sequence 0001 2 time execution of FIR sequence 0010 3 time execution of FIR sequence 0011 4 time execution of FIR sequence 0100 5 time execution of FIR sequence</p>

Table continues on the next page...

**IFC\_NCFGR field descriptions (continued)**

Field	Description
	0101 6 time execution of FIR sequence 0110 7 time execution of FIR sequence 0111 8 time execution of FIR sequence 1000 9 time execution of FIR sequence 1001 10 time execution of FIR sequence 1010 11 time execution of FIR sequence 1011 12 time execution of FIR sequence 1100 13 time execution of FIR sequence 1101 14 time execution of FIR sequence 1110 15 time execution of FIR sequence 1111 16 time execution of FIR sequence
20–23 -	This field is reserved.
24–31 NUM_WAIT	Number of wait cycles It represents the count value for which FCM will wait when used with opcode NWAIT in FIR. The value specified in this register represents the number of IFC module input clock cycles. Minimum value to be programmed is 2.

**25.3.36 NAND Flash Command register 0 (IFC\_NAND\_FCR0)**

The NAND flash command registers hold up to 8 NAND flash EEPROM command bytes that may be referenced by opcodes in NAND\_FIR during FCM operation. The values of the commands should follow the manufacturer's datasheet for the relevant NAND flash.

Address: 153\_0000h base + 1014h offset = 153\_1014h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																																	
W																	CMD0		CMD1		CMD2		CMD3										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**IFC\_NAND\_FCR0 field descriptions**

Field	Description
0–7 CMD0	General purpose FCM flash command byte 0. Opcodes in FIR that issue command index 0 write CMD0 to the NAND flash command/data bus.
8–15 CMD1	General purpose FCM flash command byte 1. Opcodes in FIR that issue command index 1 write CMD1 to the NAND flash command/data bus.
16–23 CMD2	General purpose FCM flash command byte 2. Opcodes in FIR that issue command index 2 write CMD2 to the NAND flash command/data bus.
24–31 CMD3	General purpose FCM flash command byte 3. Opcodes in FIR that issue command index 3 write CMD3 to the NAND flash command/data bus

### 25.3.37 NAND Flash Command register 1 (IFC\_NAND\_FCR1)

Address: 153\_0000h base + 1018h offset = 153\_1018h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMD4							CMD5							CMD6							CMD7										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### IFC\_NAND\_FCR1 field descriptions

Field	Description
0–7 CMD4	General purpose FCM flash command byte 4. Opcodes in FIR that issue command index 4 write CMD4 to the NAND flash command/data bus.
8–15 CMD5	General purpose FCM flash command byte 5. Opcodes in FIR that issue command index 5 write CMD5 to the NAND flash command/data bus.
16–23 CMD6	General purpose FCM flash command byte 6. Opcodes in FIR that issue command index 6 write CMD6 to the NAND flash command/data bus.
24–31 CMD7	General purpose FCM flash command byte 7. Opcodes in FIR that issue command index 7 write CMD7 to the NAND flash command/data bus

### 25.3.38 Flash Row Address register n (IFC\_ROWn)

ROWn registers are used for addressing the NAND flash memory. These registers are used as per the field NCFGR1[ADDR\_MODE]. ROWn register holds the row address to be issued on NAND flash interface using address phase. CSORn[RAL] field determines the number of bits to be issued to the flash from ROW register during row address phase.

Address: 153\_0000h base + 103Ch offset + (16d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### IFC\_ROWn field descriptions

Field	Description
0–31 RA	Row Address  This register holds the row address to be issued on flash during row address phase

### 25.3.39 Flash COL Address register n (IFC\_COLn)

COL $n$  registers are used for addressing the NAND FLASH memory. These registers are used as per the field NCFGR1[ADDR\_MODE]. The following holds the column address to be issued to flash during address phase. CSOR $n$ [PGS] field determines the number of bits to be issued to flash from COL register during column address phase.

For small-page size, that is, 512 Bytes, the four lsbs of ROW $n$  register is used to select the IFC buffer for data transfer to/from IFC buffer to/from flash. It indexes the page in NAND flash EEPROM at the current block, and locates the corresponding transfer buffer in the FCM buffer RAM.

The four lsbs of RA index, 1 of the 16, 1-KB buffers in the FCM buffer RAM as follows:

- 0000-The page is transferred to/from FCM buffer 0, address offsets 0x0000-0x03FF
- 0001-The page is transferred to/from FCM buffer 1, address offsets 0x0400-0x07FF
- 0010-The page is transferred to/from FCM buffer 2, address offsets 0x0800-0x0BFF
- 0011-The page is transferred to/from FCM buffer 3, address offsets 0x0C00-0x0FFF
- 0100-The page is transferred to/from FCM buffer 4, address offsets 0x1000-0x13FF
- 0101-The page is transferred to/from FCM buffer 5, address offsets 0x1400-0x17FF
- 0110-The page is transferred to/from FCM buffer 6, address offsets 0x1800-0x1BFF
- 0111-The page is transferred to/from FCM buffer 7, address offsets 0x1C00-0x1FFF
- 1000-The page is transferred to/from FCM buffer 8, address offsets 0x2000-0x23FF
- 1001-The page is transferred to/from FCM buffer 9, address offsets 0x2400-0x27FF
- 1010-The page is transferred to/from FCM buffer 10, address offsets 0x2800-0x2BFF
- 1011-The page is transferred to/from FCM buffer 11, address offsets 0x2C00-0x2FFF
- 1100-The page is transferred to/from FCM buffer 12, address offsets 0x3000-0x33FF
- 1101-The page is transferred to/from FCM buffer 13, address offsets 0x3400-0x37FF
- 1110-The page is transferred to/from FCM buffer 14, address offsets 0x3800-0x3BFF
- 1111-The page is transferred to/from FCM buffer 15, address offsets 0x3C00-0x3FFF

Address: 153\_0000h base + 1044h offset + (16d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R W	MS	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R W	Reserved			CA													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_COLn field descriptions**

Field	Description
0 MS	Main/spare region locator.: <b>NOTE:</b> In the case that NAND_BC[BC] = 0, MS is treated as 0. 0 Data is transferred to/from the main region of the SRAM buffer; that is, the first 512 bytes of the buffer are used. 1 Data is transferred to/from the spare region of the SRAM buffer; that is, the second 512 bytes of the buffer are used, but only an initial 16 bytes of spare region are defined.
1–18 -	This field is reserved.
19–31 CA	Column Address This register holds the column address to be issued on flash during column address phase. CA indexes the first byte to transfer to/from the main or spare region of the NAND flash EEPROM and corresponding transfer buffer. In the case that NAND_BC[BC] = 0, CA is treated as 0. For MS = 0, CA can range 0x0000-0x1FF; for MS = 1, CA can range 0x000-0x00F. For a 16-bit port size, the least significant bit of the Column Address is assumed to be zero; hence, the Column Address remains a byte index at all times.

### 25.3.40 Flash COL Address register for 2 KB Large-Page Device (IFC\_COLn\_2KB)

For large-page size of 2 KB, two lsbs of ROWn register are used to select the IFC buffer for data transfer to/from IFC buffer to/from flash. It indexes the page in NAND flash EEPROM at the current block, and locates the corresponding transfer buffer in the FCM buffer RAM.

The two lsbs of RA index one of the four 4-KB buffers in the FCM buffer RAM as follows:

- 00-The page is transferred to/from FCM buffer 0, address offsets 0x0000-0x0FFF
- 01-The page is transferred to/from FCM buffer 1, address offsets 0x1000-0x1FFF
- 10-The page is transferred to/from FCM buffer 2, address offsets 0x2000-0x2FFF
- 11-The page is transferred to/from FCM buffer 3, address offsets 0x3000-0x3FFF

Address: 153\_0000h base + 1044h offset + (16d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R W	MS	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R W	Reserved			CA													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_COLn\_2KB field descriptions**

Field	Description
0 MS	Main/spare region locator. In the case that NAND_BC[BC] = 0, MS is treated as 0.  0 Data is transferred to/from the main region of the SRAM buffer; that is, the first 2048 bytes of the buffer are used. 1 Data is transferred to/from the spare region of the SRAM buffer; that is, the second 2048 bytes of the buffer are used, but only an initial 64 bytes of spare region are defined.
1–18 -	This field is reserved.
19–31 CA	Column Address  This register holds the column address to be issued on flash during column address phase.  CA indexes the first byte to transfer to/from the main or spare region of the NAND flash EEPROM and corresponding transfer buffer. In the case that NAND_BC[BC] = 0, CA is treated as 0.  For MS = 0, CA can range 0x000-0x7FF; for MS = 1, CA can range 0x000-0x03F.  For a 16-bit port size, the least significant bit of the Column Address is assumed to be zero; hence, the Column Address remains a byte index at all times.

**25.3.41 Flash COL Address register for 4 KB Large-Page Device (IFC\_COLn\_4KB)**

For large-page size of 4 KB, lsb of ROWn register is used to select the IFC buffer for data transfer to/from IFC buffer to/from flash. It indexes the page in NAND flash EEPROM at the current block, and locates the corresponding transfer buffer in the FCM buffer RAM.

The lsb of RA indexes one of the two 8 KB buffers in the FCM buffer RAM as follows:

- 0-The page is transferred to/from FCM buffer 0, address offsets 0x0000-0x1FFF
- 1-The page is transferred to/from FCM buffer 1, address offsets 0x2000-0x3FFF

Address: 153\_0000h base + 1044h offset + (16d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	MS								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved								CA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_COLn\_4KB field descriptions**

Field	Description
0 MS	Main/spare region locator. In the case that NAND_BC[BC] = 0, MS is treated as 0.

*Table continues on the next page...*

**IFC\_COLn\_4KB field descriptions (continued)**

Field	Description
	0 Data is transferred to/from the main region of the SRAM buffer; that is, the first 4096 bytes of the buffer are used. 1 Data is transferred to/from the spare region of the SRAM buffer; that is, the second 4096 bytes of the buffer are used, but only an initial 64 bytes of spare region are defined.
1–18 -	This field is reserved.
19–31 CA	Column Address  This register holds the column address to be issued on flash during column address phase.  CA indexes the first byte to transfer to/from the main or spare region of the NAND flash EEPROM and corresponding transfer buffer. In the case that NAND_BC[BC] = 0, COL is treated as 0. For MS = 0, CA can range 0x000-0xFFFF; for MS = 1, CA can range 0x000- CSORn[SPRZ].  For a 16-bit port size, the least significant bit of the Column Address is assumed to be zero; hence, the Column Address remains a byte index at all times.

**25.3.42 Flash COL Address register for 8 KB Large-Page Device (IFC\_COLn\_8KB)**

For large-page size of 8 KB, lsb of ROWn register is used to select the IFC buffer for data transfer to/from IFC buffer to/from flash. It indexes the page in NAND flash EEPROM at the current block, and locates the corresponding transfer buffer in the FCM buffer RAM.

The lsb of RA indexes one of the two 8 KB buffers in the FCM buffer RAM as follows:

- 0-The page is transferred to/from FCM buffer 0, address offsets 0x0000-0x1FFF
- 1-The page is transferred to/from FCM buffer 1, address offsets 0x2000-0x3FFF
- Only single 8 KB page can be accessed using SRAM buffer.

Address: 153\_0000h base + 1044h offset + (16d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	MS								Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	Reserved								CA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_COLn\_8KB field descriptions**

Field	Description
0 MS	Main/spare region locator. In the case that NAND_BC[BC] = 0, MS is treated as 0.

*Table continues on the next page...*

**IFC\_COLn\_8KB field descriptions (continued)**

Field	Description
	<p>0 Data is transferred to/from the main region of the SRAM buffer; that is, the first 4096 bytes of the buffer are used.</p> <p>1 Data is transferred to/from the spare region of the SRAM buffer; that is, the second 8192 bytes of the buffer are used, but only an initial CSOR[SPRZ] bytes of spare region are defined. Maximum addressable bytes in sprae region buffer are 1024 bytes with this page size.</p>
1–18 -	This field is reserved.
19–31 CA	<p>Column Address</p> <p>This register holds the column address to be issued on Flash during column address phase. CA indexes the first byte to transfer to/from the main or spare region of the NAND flash E2PROM and corresponding transfer buffer. In the case that NAND_BC[BC] = 0, COL is treated as 0. For MS = 0, CA can range 0x000-0x1FFF; for MS = 1, CA can range 0x000- CSOR[SPRZ]. For a 16 bit port size or a NVDDR device the least significant bit of the column address is assumed to be zero, hence, the column address remains a byte index at all times.</p>

**25.3.43 Flash Byte Count register for NAND Flash (IFC\_NAND\_BC)**

This register defines the NAND flash data block transfer size.

Address: 153\_0000h base + 1108h offset = 153\_1108h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
W	Reserved															BC																

**IFC\_NAND\_BC field descriptions**

Field	Description
0–17 -	This field is reserved.
18–31 BC	<p>Byte count determines how many bytes are transferred by the flash controller during data read (RBCD) or data write (WBCD) opcodes. In case of partial page operations, the value programmed in this field must be aligned to the port size of the device. Thus, if the bank port size is 16 bits, the lsb bit of BC (that is, bit 31) is ignored.</p> <p>The first byte accessed in the NAND flash is located by the COLn register, and successive bytes are transferred until BC bytes have been counted</p> <p>If BC = 0, an entire flash page and its spare region will be transferred by FCM, in which case COLn[MS] and COLn[CA] are treated as zero regardless of their values.</p> <p><b>NOTE:</b> BC = 0 is the only setting that permits flash controller to generate and check ECC.</p>

### 25.3.44 NAND Flash Instruction register 0 (IFC\_NAND\_FIR0)

The following flash instruction registers hold the instructions. When any operation on a selected bank is triggered and the FIR is programmed by the user, flash controller executes 6-bit opcode at a time until all the opcodes have been fetched from all the FIR registers. There are three such registers provided for the application. A total of 15 instructions can be programmed for FLASH operations.

NAND flash instruction register0 holds first five opcodes.

**Table 25-5. Instruction Opcodes**

Opcode encoding (6 bits)	Opcode name	Description
0x00	NOOP	No-operation and end of operation sequence
0x01	CA0	Issue current column address (CA) as set in COL0
0x02	CA1	Issue current column address as set in COL1
0x03	CA2	Issue current column address as set in COL2
0x04	CA3	Issue current column address as set in COL3
0x05	RA0	Issue current row address as set in ROW0
0x06	RA1	Issue current row address as set in ROW1
0x07	RA2	Issue current row address as set in ROW2
0x08	RA3	Issue current row address as set in ROW3
0x09	CMD0	Issue command from NAND_FCR0[CMD0]
0x0A	CMD1	Issue command from NAND_FCR0[CMD1]
0x0B	CMD2	Issue command from NAND_FCR0[CMD2]
0x0C	CMD3	Issue command from NAND_FCR0[CMD3]
0x0D	CMD4	Issue command from NAND_FCR1[CMD4]
0x0E	CMD5	Issue command from NAND_FCR1[CMD5]
0x0F	CMD6	Issue command from NAND_FCR1[CMD6]
0x10	CMD7	Issue command from NAND_FCR1[CMD7]
0x11	CW0	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR0[CMD0]
0x12	CW1	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR0[CMD1]
0x13	CW2	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR0[CMD2]
0x14	CW3	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR0[CMD3]

*Table continues on the next page...*

**Table 25-5. Instruction Opcodes (continued)**

Opcode encoding (6 bits)	Opcode name	Description
0x15	CW4	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR1[CMD4]
0x16	CW5	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR1[CMD5]
0x17	CW6	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR1[CMD6]
0x18	CW7	Wait for TWBE time, poll R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then issue command from NAND_FCR1[CMD7]
0x19	WBCD	Write BC bytes of data from current FCM buffer to flash device
0x1A	RBCD	Wait for R/B_B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then Read BC bytes of data from flash device into current FCM RAM buffer
0x1B	BTRD	Same as RBCD except in this case deassertion of read enable will happen once the read data has been sampled (same as boot read)
0x1C	RDSTAT	Wait for TWHR (for NVDDR)/TWHRE (Async) time, then Read one byte/two bytes (8b/16b port) of data from flash device into RS0 and RS1 field of FSR.
0x1D	NWAIT	Wait for NCFG[NUM_WAIT] Clock cycles
0x1E	WFR	Wait for TWBE time, poll Ready
0x1F	SBRD	Wait for R/B to return high or time-out (depends upon IFC_NANDCR[FTOCNT]), then Read 8/16 bits of data from NAND flash memory into the NAND_MDR register. COLn register will determine the location from where the data will be fetched in a given page.  Deassertion of read enable will happen once the read data has been sampled.  With this opcode NAND_BC register field value will be ignored. Although the value is ignored, it is mandatory that the NAND_BC register hold a non-zero value else the Column Address would be treated as 0 (refer <a href="#">Flash COL Address register n (IFC_COLn)</a> )
0x20	UA	Issue current row address as set in ROW3  The SRAM Buffer used for data transfers is always Buffer 0
0x21	RB	Wait for TWHRE (for NVDDR)/TWHRE (Async) time, then Read BC bytes of data from flash device into current FCM RAM buffer. Deassertion of read enable will happen once the read data has been sampled
Others	-	Reserved

Address: 153\_0000h base + 1110h offset = 153\_1110h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## IFC memory map/register definition

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	OP2				OP3				OP4				Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### IFC\_NAND\_FIR0 field descriptions

Field	Description
0–5 OP0	Opcode0
6–11 OP1	Opcode1
12–17 OP2	Opcode2
18–23 OP3	Opcode3
24–29 OP4	Opcode4
30–31 -	This field is reserved.

## 25.3.45 NAND Flash Instruction register 1 (IFC\_NAND\_FIR1)

NAND flash instruction register1 holds other five opcodes.

Address: 153\_0000h base + 1114h offset = 153\_1114h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	OP5				OP6				OP7				OP8			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	OP7		OP8				OP9				Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### IFC\_NAND\_FIR1 field descriptions

Field	Description
0–5 OP5	Opcode5
6–11 OP6	Opcode6
12–17 OP7	Opcode7
18–23 OP8	Opcode8

Table continues on the next page...

**IFC\_NAND\_FIR1 field descriptions (continued)**

Field	Description
24–29 OP9	Opcode9
30–31 -	This field is reserved.

**25.3.46 NAND Flash Instruction register 2 (IFC\_NAND\_FIR2)**

NAND flash instruction register2 holds last five opcodes.

Address: 153\_0000h base + 1118h offset = 153\_1118h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	OP10						OP11						OP12			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	OP12		OP13						OP14						Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_NAND\_FIR2 field descriptions**

Field	Description
0–5 OP10	Opcode10
6–11 OP11	Opcode11
12–17 OP12	Opcode12
18–23 OP13	Opcode13
24–29 OP14	Opcode14
30–31 -	This field is reserved.

### 25.3.47 NAND Chip-Select register (IFC\_NAND\_CSEL)

This register tells the NAND FCM about the selected chip-select on which a program or read operation has to be performed. As NAND flash is accessed through an SRAM buffer (memory-mapped), software tells the NAND FCM which chip-select is desired using this register.

Address: 153\_0000h base + 115Ch offset = 153\_115Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	CSEL															Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### IFC\_NAND\_CSEL field descriptions

Field	Description
0–2 -	This field is reserved.
3–5 CSEL	Chip-Select for NAND flash operation  000 Chip-select0 001 Chip-select1 010 Chip-select2 011 Chip-select3  ... 110 Chip select6 111 Reserved
6–31 -	This field is reserved.

### 25.3.48 NAND Operation Sequence Start (IFC\_NANDSEQ\_STRT)

This register is used to trigger the operation on NAND flash.

The following figure shows the NANDSEQ\_STRT register. Only one operation can be triggered at a time. Software has to trigger the operation and the chip will clear this after completing the operation.

Address: 153\_0000h base + 1164h offset = 153\_1164h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	NAND_FIR_STRT								AUTO_ERS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	AUTO_RD		Reserved	AUTO_STAT_RD											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_NANDSEQ\_STRT field descriptions**

Field	Description
0 NAND_FIR_STRT	NAND flash operation start Writing 1 to this register bit triggers normal operation sequence programmed in NAND FIR registers on NAND flash.
1–7 -	This field is reserved.
8 AUTO_ERS	Automatic erase Writing 1 to this register bit triggers automatic erase operation <b>NOTE:</b> The FIR sequence executed is {CW0, RA0, CMD1, and NOOP}
9–10 -	This field is reserved.
11 AUTO_PGM	Automatic program Writing 1 to this register bit triggers automatic program operation <b>NOTE:</b> The FIR sequence executed is {CW0, CA0, RA0, WBCD, CMD1, and NOOP}. This sequence will not support small page program, hence user cannot use auto program for small page device.
12–13 -	This field is reserved.
14 AUTO_CPB	Automatic copyback Writing 1 to this register bit triggers automatic copy back operation <b>NOTE:</b> The FIR sequence executed is {CW0, CA0, RA0, CMD1, CW2, CA1, RA1, CMD3, CW4, RDSTAT, and NOOP}
15–16 -	This field is reserved.
17 AUTO_RD	Automatic read operation Writing 1 to this register bit triggers automatic read operation <b>NOTE:</b> The FIR sequence executed is:

*Table continues on the next page...*

**IFC\_NANDSEQ\_STRT field descriptions (continued)**

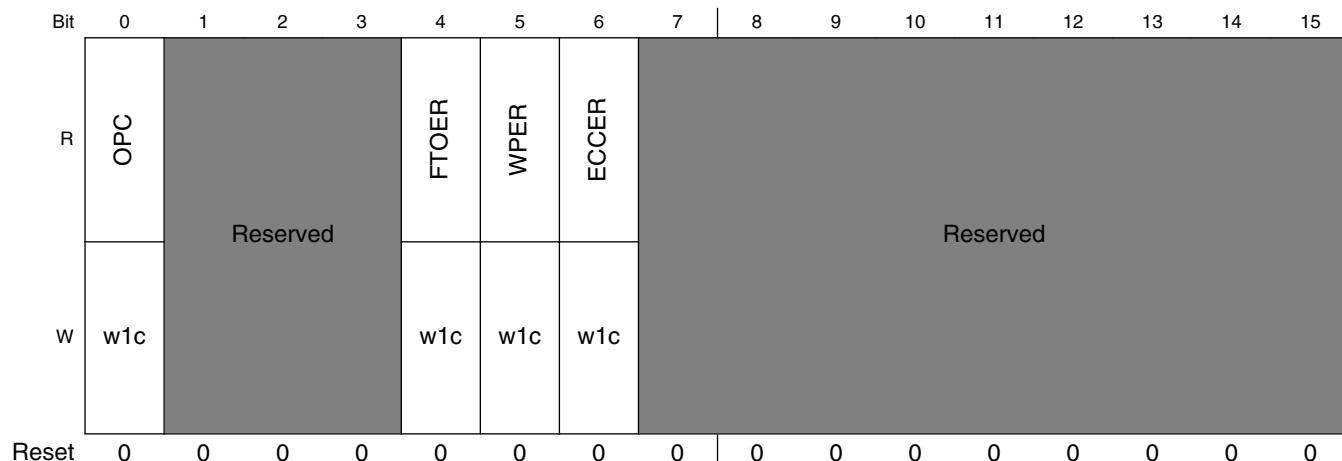
Field	Description
	For small-page device: {CW0, CA0, RA0, RBCD, NOOP} For large-page device: {CW0, CA0, RA0, CMD1, RBCD, NOOP}
18–19 -	This field is reserved.
20 AUTO_STAT_RD	Automatic status read Writing 1 to this register bit triggers automatic read status <b>NOTE:</b> The FIR sequence executed is {CW0, RDSTAT, NOOP}
21–31 -	This field is reserved.

### 25.3.49 NAND Event and Error Status register (IFC\_NAND\_EVTER\_STAT)

NAND event and error status register (NAND\_EVTER\_STAT) indicates the cause of an error or event corresponding to NAND flash.

The following figure shows the register fields. It is write-1-to-clear register.

Address: 153\_0000h base + 116Ch offset = 153\_116Ch



Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RCW_DN	BOOT_DN			BBL_SRCH_SEL											
W	w1c	w1c	Reserved		w1c											

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

**IFC\_NAND\_EVTER\_STAT field descriptions**

Field	Description
0 OPC	<p>NAND flash operation complete event. OPC indicates that all instructions in FIR has been executed.</p> <p><b>NOTE:</b> 1. The software should not poll this bit during auto boot. Instead RCW_DN/ BOOT_DN should be used. However, software must clear this bit post auto boot.</p> <p><b>NOTE:</b> 2. This bit is set even if a flash timeout, ECC or write protect error occurs. For example, if a flash timeout error is detected for a read operation initiated on the NAND flash memory device, IFC will dump BC bytes worth of garbage data from the NAND flash interface into the internal SRAM and set this bit. Similarly, in case of NAND write protect error, IFC asserts chip select and sends the program data to NAND device, it is device which ignores the data as write protect signal is asserted. User must wait for operation completion (poll for OPC bit getting set) even in case of timeout before initiating next operation.</p> <p>As the transaction goes into NAND, OPC will be set at the end of FIR execution. OPC bit indicates that all the instructions in the FIRs have been executed. Also, it implies that OPC should be polled first and then FSR for completion.</p> <p>0 NAND flash operation is not complete 1 NAND flash operation completed</p>
1–3 -	This field is reserved.
4 FTOER	<p>Flash timeout error</p> <p>0 No flash interface timeout 1 Flash interface timeout occurred</p>
5 WPER	<p>Write protect error</p> <p><b>NOTE:</b> Error is asserted when the write operation starts executing on the flash interface.</p> <p>0 No write protect error 1 A write is attempted to the memory bank which is write protected</p>
6 ECCER	<p>ECC error</p> <p><b>NOTE:</b> With the clearing of this error status bit, the ECCSTAT0/1/2/3 register contents will also be cleared.</p>

*Table continues on the next page...*

**IFC\_NAND\_EVTER\_STAT field descriptions (continued)**

Field	Description
	0 No uncorrectable ECC Error occurred on page read operation 1 Uncorrectable ECC error occurred in page read operation
7–15 -	This field is reserved.
16 RCW_DN	This bit is set when one flash page has been read from the flash device and written into the SRAM buffer during RCW load. It is write-1-to-clear bit and can be cleared by software after RCW loading. This event does not have any corresponding event enable and interrupt enable bit.
17 BOOT_DN	This event is set when 8 KB flash data has been read from the flash device and written into the SRAM buffer during BOOT load. It is write-1-to-clear bit and can be cleared by software after BOOT loading. This event will not have any corresponding event enable and interrupt enable bit.
18–19 -	This field is reserved.
20 BBI_SRCH_SEL	Bad Block Indicator search select: This status register bit represents the location of bad block indication in each block of NAND flash device connected at chip-select0. This field will be updated with input port value por_cfg_bbi_srch_sel when por_cfg_boot_load /por_cfg_rcw_load pulse occurs. 0 Read bad block indicator (BBI) corresponding to page0 and page1 of each block of the NAND flash device connected at chipselect 0, to identify the first good block during auto-boot 1 Read bad block indicator (BBI) corresponding to page0 and the last page of each block of the NAND flash device connected at chipselect 0, to identify the first good block during auto-boot
21–31 -	This field is reserved.

### 25.3.50 NAND Page Read Completion Event Status register (IFC\_PGRDCMPL\_EVT\_STAT)

NAND flash page read completion event register indicates the status of the pages read from NAND flash and stored in SRAM buffer. 16 bits of this register represent 16 sectors each of 512 bytes (used in ECC decoding). There can be 1, 4, 8, and 16 sectors in each page of size 512 bytes, 2 KB, 4 KB, and 8 KB . Event interrupt will be generated if one complete page has been written in SRAM buffer (when ecc\_dec\_en = 0) or when one complete page has been fixed in the SRAM after decoding (ecc\_dec\_en = 1). Software can read the corresponding page whose data has been completely written in SRAM (ECC fixed) and clear the corresponding bits in this register.

24/40-bit ECC is computed on 1 KB sector and applicable for 4 KB and 8 KB page sizes. However the definition of this register remains same except the fact that 512 is the byte size and not the sector size.

#### NOTE

The register gets cleared when software clears the OPC bit in NAND\_EVTER\_STAT register. Even if per page event interrupt is used for reading the data from SRAM,

NAND\_EVTER\_STAT[OPC] is the true indication of NAND operation completion.

The following figure shows the register fields. It is write-1-to-clear register.

Address: 153\_0000h base + 1174h offset = 153\_1174h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
	SEC_DONE															Reserved																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

### IFC\_PGRDCMPL\_EVT\_STAT field descriptions

Field	Description
0–15 SEC_DONE	<p>For small page: Each of the 16 bits of this field represent one small page read data completion status</p> <p>Bit 0 set-Page 0 done, that is, flash read data is completely written and fixed in SRAM buffer</p> <p>Bit 1 set-Page 1 done</p> <p>Bit 2 set-Page 2 done</p> <p>Bit 3 set-Page 3 done</p> <p>...</p> <p>Bit15 Set-Page 15 done</p> <p>For 2KB Page: SRAM buffer can accommodate 4 page (2KB) each containing four 512 byte sectors</p> <p>SEC_DONE[0:3] = 4'b1111: Page 0 done</p> <p>SEC_DONE[4:7] = 4'b1111: Page 1 done</p> <p>SEC_DONE[8:11] = 4'b1111: Page 2 done</p> <p>SEC_DONE[12:15] = 4'b1111: Page 3 done</p> <p>For 4KB Page: SRAM buffer can accommodate 2 page (4KB) each containing eight 512 byte sectors</p> <p>SEC_DONE[0:7] = 8'b1111_1111: Page 0 done</p> <p>SEC_DONE[8:15] = 8'b1111_1111: Page 1 done</p> <p>For 8KB Page: Only one page can be buffered in SRAM.</p> <p>SEC_DONE[0:15] = 16'b 1111_1111_1111_1111: Single page of 8KB done.</p> <p><b>NOTE:</b> Event interrupt will be generated when one complete page read is done. Software has to read the corresponding page from SRAM buffer and clear the bits from this register. For example if page size is 2 KB and page 0 is being written in SRAM buffer, then SEC_DONE[0:15] will be 16'HF000, now software has to write the data 16'HF000 in this register to clear it.</p>
16–31 -	This field is reserved.

### 25.3.51 NAND Event and Error Enable register (IFC\_NAND\_EVTER\_EN)

Event and Error Enable Register (NAND\_EVTER\_EN) is used to enable/disable the logging of event and error indication in the NAND\_EVTER\_STAT and PGRDCMPL\_EVT\_STAT registers.

Address: 153\_0000h base + 1180h offset = 153\_1180h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	OPCEN	Reserved	PGRDCMPLEN	Reserved	FTOEREN	WPEREN	ECCEREN									
W																
Reset	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_NAND\_EVTER\_EN field descriptions

Field	Description
0 OPCEN	NAND flash operation complete event enable 0 NAND flash operation complete event is disabled 1 NAND flash operation completed event is enabled
1 -	This field is reserved.
2 PGRDCMPLEN	NAND flash page read completion event enable 0 Per page read event disable 1 Event will be generated on per page flash read operation completion. Status about the page whose data is available in sram buffer is given by PGRDCMPL_EVT_STAT register.

Table continues on the next page...

**IFC\_NAND\_EVTER\_EN field descriptions (continued)**

Field	Description
3 -	This field is reserved.
4 FTOEREN	Flash time out error enable (for R/B timeout and DQS timeout) 0 Flash timeout error is disabled 1 Flash timeout error is enabled
5 WPEREN	Write protect error checking enable. 0 No write protect error checking 1 Write protect error checking is enabled
6 ECCEREN	ECC error logging enable. 0 ECCER event is not logged in NAND_EVTER_STAT register 1 ECCER event is logged in NAND_EVTER_STAT register
7–31 -	This field is reserved.

### 25.3.52 NAND Event and Error Interrupt Enable register (IFC\_NAND\_EVTER\_INTR\_EN)

Event and Error Interrupt Enable Register (NAND\_EVTER\_INTR\_EN) is used to enable/disable the generation of interrupt signals corresponding to error and event reporting. Software must clear pending events and errors in NAND\_EVTER\_STAT and PGRDCMPL\_EVT\_STAT register before enabling the interrupts.

Address: 153\_0000h base + 118Ch offset = 153\_118Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	OPCIREN	Reserved	PGRDCMPLIREN	Reserved	FTOERIREN	WPERIREN	ECCERIREN									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_NAND\_EVTER\_INTR\_EN field descriptions

Field	Description
0 OPCIREN	NAND flash operation complete event interrupt enable 0 NAND flash operation Complete Event Interrupt is disabled 1 NAND flash operation completed Event Interrupt is Enabled
1 -	This field is reserved.
2 PGRDCMPLIREN	Page read completion event interrupt enable 0 NAND flash page read event interrupt is disabled (per page basis) 1 NAND flash page read event interrupt is enabled

Table continues on the next page...

**IFC\_NAND\_EVTER\_INTR\_EN field descriptions (continued)**

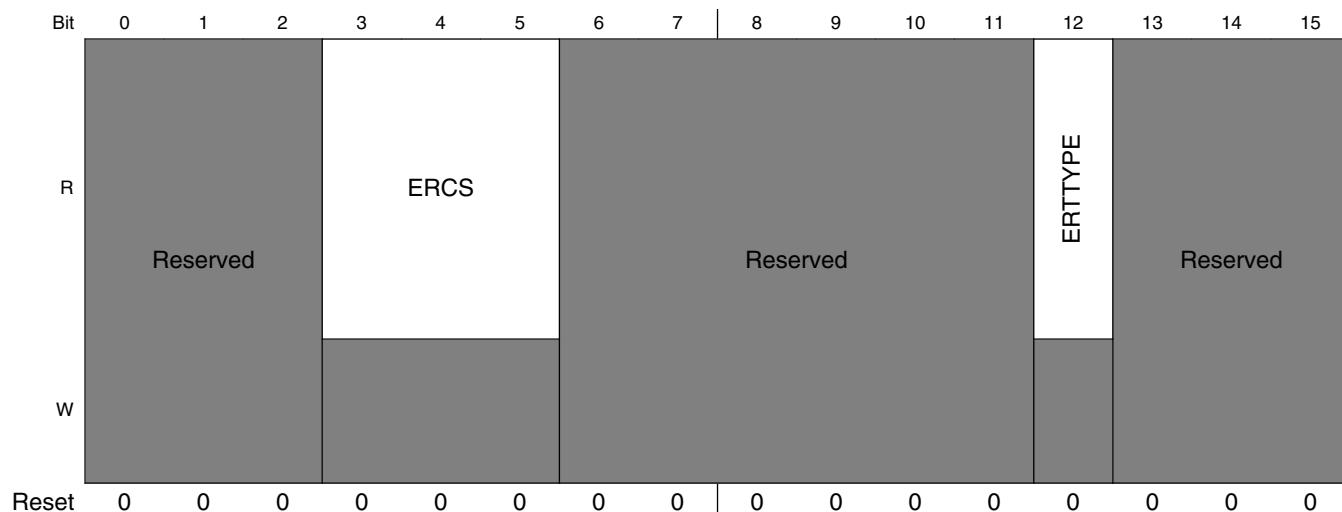
Field	Description
3 -	This field is reserved.
4 FTOERIREN	Flash timeout error interrupt enable (for R/B timeout and DQS timeout) 0 Flash timeout error interrupt is disabled 1 Flash timeout error interrupt is enabled
5 WPERIREN	Write protect error interrupt enable 0 Write protect error interrupt disable 1 Write protect error interrupt is enabled
6 ECCERIREN	ECC error interrupt enable 0 ECC Error Interrupt is disabled 1 ECC Error Interrupt is enabled
7-31 -	This field is reserved.

### 25.3.53 NAND Transfer Error Attributes register 0 (IFC\_NAND\_ERATTR0)

Transfer error attribute register 0 (NAND\_ERATTR0) is used to register the transaction attributes corresponding to the first error.

These registers store the attribute corresponding to the first transaction on which an error occurred.

Address: 153\_0000h base + 1198h offset = 153\_1198h



## IFC memory map/register definition

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### IFC\_NAND\_ERATTR0 field descriptions

Field	Description
0–2 -	This field is reserved.
3–5 ERCS	Chip-Select Corresponding to NAND Error  000 Bank0 001 Bank1 010 Bank2  ... 111 Reserved
6–11 -	This field is reserved.
12 ERTTYPE	Transaction type of NAND error  0 Write 1 Read
13–31 -	This field is reserved.

### 25.3.54 NAND Transfer Error Attributes register 1 (IFC\_NAND\_ERATTR1)

Transfer error attribute register (NAND\_ERATTR1) is used to register the transaction address corresponding to the first error.

Address: 153\_0000h base + 119Ch offset = 153\_119Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_NAND\_ERATTR1 field descriptions**

Field	Description
0–31 ER_ROWAD	Row address corresponding to error transaction

**25.3.55 NAND Flash Status register (IFC\_NAND\_FSR)**

Flash status register (NAND\_FSR) contains read status data from NAND flash.

Address: 153\_0000h base + 11E0h offset = 153\_11E0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RS0							RS1							Reserved																	
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**IFC\_NAND\_FSR field descriptions**

Field	Description
0–7 RS0	First byte of data read from read status operation
8–15 RS1	Second byte of data read from read status operation
16–31 -	This field is reserved.

**25.3.56 ECC Status and Result of Flash Operation register 0 (IFC\_ECCSTAT0)**

ECC status registers are used to store the number of ECC errors occurred on read operation. It has different meanings for 4/8 and 24/40 bit ECC Modes as the sector size is different for 4/8 and 24/40 bit modes.

ECC is computed on sector basis, and in the SRAM buffer there can be 16 such sectors of 512 byte each for 4 and 8 bit ECC (Galois field GF- $2^{13}$ ) and 8 sectors of 1 KB each for 24 and 40 bit ECC (Galois Field 2 $^{14}$ ). Hence, each field of these registers represents number of errors in each sector. Depending on the page index mapped in the SRAM buffer, the corresponding field in the ECCSTAT0 will be updated.

As SRAM buffer can have 8 sectors of 1KB each, only the first eight register fields in ECCSTAT0/1 registers are defined with width 6 to represent 24 and 40 bit ECC along with 4 and 8 bit ECC.

**NOTE**

1. These registers will be updated whenever decoder is enable (ECC\_DEC\_EN =1'b1) irrespective of ECC Error Event Indication is enabled or not (ECCEREN).
2. With the clearing of NAND\_EVTER\_STAT[ECCER] register bit the contents of ECCSTAT0/1/2/3 registers will also be cleared. These registers will also be cleared with soft\_reset.
3. For every NAND sequence start, all the ECCSTAT registers will be cleared. It is implemented in order to flush the previously logged information.

Address: 153\_0000h base + 11E8h offset = 153\_11E8h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		NUMERO						Reserved		NUMER1					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		NUMER2						Reserved		NUMER3					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_ECCSTAT0 field descriptions**

Field	Description
0–1 -	This field is reserved.
2–7 NUMERO	<p>Number of ECC errors on sector #0 of SRAM buffer</p> <p>For 4/8 bit ECC Mode (512 byte sector):</p> <ul style="list-style-type: none"> <li>000000 No Error</li> <li>000001 1 Bit Error</li> <li>000010 2 Bit Error</li> <li>000011 3 bit Error</li> <li>000100 4 Bit Error</li> <li>000101 5 Bit Error</li> <li>000110 6 Bit Error</li> <li>000111 7 Blt Error</li> <li>001000 8 Bit Error</li> <li>001111 - Uncorrectable Errors</li> <li>Others Reserve</li> </ul> <p>For 24/40 bit ECC Mode (1KB sector):</p>

Table continues on the next page...

**IFC\_ECCSTAT0 field descriptions (continued)**

Field	Description
	000000 No Error 000001 - 101000 1 Bit Error to 40 bit Error in this sector 111111 Uncorrectable Errors Others Reserve
8–9 -	This field is reserved.
10–15 NUMER1	Number of ECC errors on sector #1 of SRAM buffer
16–17 -	This field is reserved.
18–23 NUMER2	Number of ECC errors on sector #2 of SRAM buffer
24–25 -	This field is reserved.
26–31 NUMER3	Number of ECC errors on sector #3 of SRAM buffer

**25.3.57 ECC Status and Result of Flash Operation register 1 (IFC\_ECCSTAT1)**

ECC Status Register (ECCSTAT1) is used to store the number of ECC errors occurred on Sector 4-7 during page read operation.

Address: 153\_0000h base + 11ECh offset = 153\_11ECh

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_ECCSTAT1 field descriptions**

Field	Description
0–1 -	This field is reserved.

*Table continues on the next page...*

**IFC\_ECCSTAT1 field descriptions (continued)**

Field	Description
2–7 NUMER4	Number of ECC errors on sector #4 of SRAM buffer
8–9 -	This field is reserved.
10–15 NUMER5	Number of ECC errors on sector #5 of SRAM buffer
16–17 -	This field is reserved.
18–23 NUMER6	Number of ECC errors on sector #6 of SRAM buffer
24–25 -	This field is reserved.
26–31 NUMER7	Number of ECC errors on sector #7 of SRAM buffer

### 25.3.58 ECC Status and Result of Flash Operation register 2 (IFC\_ECCSTAT2)

ECC Status Register (ECCSTAT1) is used to store the number of ECC errors occurred on Sector 8-11 during page read operation.

Address: 153\_0000h base + 11F0h offset = 153\_11F0h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				NUMER8				Reserved				NUMER9				Reserved				NUMER10				Reserved				NUMER11			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_ECCSTAT2 field descriptions**

Field	Description
0–3 -	This field is reserved.
4–7 NUMER8	Number of ECC errors on sector #8 of SRAM buffer
8–11 -	This field is reserved.
12–15 NUMER9	Number of ECC errors on sector #9 of SRAM buffer
16–19 -	This field is reserved.

Table continues on the next page...

**IFC\_ECCSTAT2 field descriptions (continued)**

Field	Description
20–23 NUMER10	Number of ECC errors on sector #10 of SRAM buffer
24–27 -	This field is reserved.
28–31 NUMER11	Number of ECC errors on sector #11 of SRAM buffer

### 25.3.59 ECC Status and Result of Flash Operation register 3 (IFC\_ECCSTAT3)

ECC Status Register (ECCSTAT3) is used to store the number of ECC errors occurred on Sector 12-15 during page read operation.

Address: 153\_0000h base + 11F4h offset = 153\_11F4h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				NUMER12		Reserved			NUMER13		Reserved			NUMER14		Reserved			NUMER15												
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_ECCSTAT3 field descriptions**

Field	Description
0–3 -	This field is reserved.
4–7 NUMER12	Number of ECC errors on sector #12 of SRAM buffer
8–11 -	This field is reserved.
12–15 NUMER13	Number of ECC errors on sector #13 of SRAM buffer
16–19 -	This field is reserved.
20–23 NUMER14	Number of ECC errors on sector #14 of SRAM buffer
24–27 -	This field is reserved.
28–31 NUMER15	Number of ECC errors on sector #15 of SRAM buffer

### 25.3.60 NAND Control register (IFC\_NANDCR)

Address: 153\_0000h base + 1278h offset = 153\_1278h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved			FTOCNT													Reserved															
Reset	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### IFC\_NANDCR field descriptions

Field	Description																																
0–2 -	This field is reserved.																																
3–6 FTOCNT	<p>Flash timeout count</p> <table> <tr><td>0000</td><td>256 cycles of IFC module input clock</td></tr> <tr><td>0001</td><td>512 cycles of IFC module input clock</td></tr> <tr><td>0010</td><td>1024 cycles of IFC module input clock</td></tr> <tr><td>0011</td><td>2048 cycles of IFC module input clock</td></tr> <tr><td>0100</td><td>4096 cycles of IFC module input clock</td></tr> <tr><td>0101</td><td>8192 cycles of IFC module input clock</td></tr> <tr><td>0110</td><td>16,384 cycles of IFC module input clock</td></tr> <tr><td>0111</td><td>32,768 cycles of IFC module input clock</td></tr> <tr><td>1000</td><td>65,536 cycles of IFC module input clock</td></tr> <tr><td>1001</td><td>131,072 cycles of IFC module input clock</td></tr> <tr><td>1010</td><td>262,144 cycles of IFC module input clock</td></tr> <tr><td>1011</td><td>524,288 cycles of IFC module input clock</td></tr> <tr><td>1100</td><td>1,048,576 cycles of IFC module input clock</td></tr> <tr><td>1101</td><td>2,097,152 cycles of IFC module input clock</td></tr> <tr><td>1110</td><td>4,194,304 cycles of IFC module input clock</td></tr> <tr><td>1111</td><td>8,388,608 cycles of IFC module input clock</td></tr> </table>	0000	256 cycles of IFC module input clock	0001	512 cycles of IFC module input clock	0010	1024 cycles of IFC module input clock	0011	2048 cycles of IFC module input clock	0100	4096 cycles of IFC module input clock	0101	8192 cycles of IFC module input clock	0110	16,384 cycles of IFC module input clock	0111	32,768 cycles of IFC module input clock	1000	65,536 cycles of IFC module input clock	1001	131,072 cycles of IFC module input clock	1010	262,144 cycles of IFC module input clock	1011	524,288 cycles of IFC module input clock	1100	1,048,576 cycles of IFC module input clock	1101	2,097,152 cycles of IFC module input clock	1110	4,194,304 cycles of IFC module input clock	1111	8,388,608 cycles of IFC module input clock
0000	256 cycles of IFC module input clock																																
0001	512 cycles of IFC module input clock																																
0010	1024 cycles of IFC module input clock																																
0011	2048 cycles of IFC module input clock																																
0100	4096 cycles of IFC module input clock																																
0101	8192 cycles of IFC module input clock																																
0110	16,384 cycles of IFC module input clock																																
0111	32,768 cycles of IFC module input clock																																
1000	65,536 cycles of IFC module input clock																																
1001	131,072 cycles of IFC module input clock																																
1010	262,144 cycles of IFC module input clock																																
1011	524,288 cycles of IFC module input clock																																
1100	1,048,576 cycles of IFC module input clock																																
1101	2,097,152 cycles of IFC module input clock																																
1110	4,194,304 cycles of IFC module input clock																																
1111	8,388,608 cycles of IFC module input clock																																
7–31 -	This field is reserved.																																

### 25.3.61 NAND Autoboot Trigger register (IFC\_NAND\_AUTOBOOT\_TRGR)

This register is used to trigger the booting on the NAND flash. This register is provided if por\_cfg\_rcw\_load and por\_cfg\_boot\_load signals are not coming through the pins for the booting. The user can use this register to achieve the same boot functionality as described in [NAND asynchronous mode boot mechanism](#) .

It is a self-clearing register; hence, the user just has to write 1 in appropriate field to trigger the auto-boot operation and hardware will clear it.

Address: 153\_0000h base + 1284h offset = 153\_1284h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R					0											
W	RCW_LD	Reserved		BOOT_LD										Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_NAND\_AUTOBOOT\_TRGR field descriptions**

Field	Description
0 RCW_LD	<p>RCW Load</p> <p><b>NOTE:</b> The values on the various por_cfg pins should be driven valid when this bit is set.</p> <p>0 No RCW loading</p> <p>1 Trigger RCW load from NAND flash (same functionality that can be achieved by sending pulse on por_cfg_rcw_load)</p>

*Table continues on the next page...*

**IFC\_NAND\_AUTOBOOT\_TRGR field descriptions (continued)**

Field	Description
1 -	This field is reserved.
2 BOOT_LD	BOOT Load <b>NOTE:</b> The values on the various por_cfg pins should be driven valid when this bit is set. 0 No BOOT loading 1 Trigger autoboot loading from NAND flash (same functionality that can be achieved by sending pulse on por_cfg_boot_load)
3–31 -	This field is reserved.

**25.3.62 NAND Flash Memory Data register (IFC\_NAND\_MDR)**

This register is used to store one beat of data read from NAND flash memory when opcode, "SBRD" is used in the FIR register. This register is provided for reading 1 or 2 bytes of data from 8- or 16-bit NAND flash. The location from where the data gets fetched can be controlled by column register (COL<sub>n</sub>). When NAND FIR is programmed with opcode "SBRD", IFC always reads 1/2 bytes of data irrespective of the value of BC (no. of bytes) programmed in NAND\_BC register.

Also note that the read data fetched from NAND flash is only stored in this register and SRAM buffer does not get updated as this opcode is intended for register access only.

Address: 153\_0000h base + 128Ch offset = 153\_128Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RDATA0								RDATA1								Reserved															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**IFC\_NAND\_MDR field descriptions**

Field	Description
0–7 RDATA0	1st read data byte when opcode SBRD is used for read
8–15 RDATA1	Second read data byte when opcode SBRD is used for read (only valid for 16 bit NAND flash)
16–31 -	This field is reserved.

### 25.3.63 Nand DLL Low Config 0 Register (IFC\_NAND\_DLL\_LOW\_CFG0)

This register is used to configure the DLL used to facilitate the shifting on the incoming DQS to the centre of the data eye during read operations. This register is used to configure the DLL for interface frequency upto 133 MHz (based on CSOR\_EXT[MODE\_FREQ]).

Address: 153\_0000h base + 1300h offset = 153\_1300h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DLL_ENABLE	DLL_RESET														
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_NAND\_DLL\_LOW\_CFG0 field descriptions**

Field	Description
0 DLL_ENABLE	This bit Enables the DLL. 0 DLL disabled. 1 DLL enabled.
1 DLL_RESET	This bit resets the DLL. When this bit is set, TAP 0 for both the master and the slave delay lines gets selected. 0 DLL is not reset. 1 DLL is reset.
2-31 -	This field is reserved.

### 25.3.64 Nand DLL Low Config 1 Register (IFC\_NAND\_DLL\_LOW\_CFG1)

This register is used to configure the DLL used to facilitate shifting of the incoming DQS to the centre of the data eye during read operations. This register is used to configure the DLL for interface frequency upto 133 Mhz (based on CSOR\_EXT[MODE\_FREQ])

Address: 153\_0000h base + 1304h offset = 153\_1304h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DLL_PD_PULSE_STRETCH_SEL															
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_NAND\_DLL\_LOW\_CFG1 field descriptions

Field	Description
0 DLL_PD_ PULSE_ STRETCH_SEL	Used to select between a 2 delay cell or 4 delays cells for reliable detection of pulse width by the pulse width detection logic.  0 4 delay cell selected. 1 2 delay cell selected.
1–11 -	This field is reserved. Reserved
12–15 DLL_REF_ UPDATE_INT	This field is used to override the default update interval of the reference delay line. The default value of the update interval is 2 REF_CLK cycles. When this field is programmed, the value of the update interval is 2 + DLL_REF_UPDATE_INT  0000 - Update interval value is 2 + 0 REF clock cycles. 0001 - Update interval value is 2 + 1 REF clock cycles. 0010 - Update interval value is 2 + 2 REF clock cycles. 0011 - Update interval value is 2 + 3 REF clock cycles.

Table continues on the next page...

**IFC\_NAND\_DLL\_LOW\_CFG1 field descriptions (continued)**

Field	Description
	... 1111 - Update interval value is 2 + 15 REF clock cycles.
16–23 -	This field is reserved. Reserved
24–31 DLL_SLV_UPDATE_INT	This field is used to override the default update interval of the slave delay line. The default value of the update interval is 256 REF_CLK cycles. This field is valid only if a non zero value is programmed. 00000000- Update interval value is 256 REF clock cycles. 00000001- Update interval value is 1 REF clock cycles. 00000010- Update interval value is 2 REF clock cycles. 00000011- Update interval value is 3 REF clock cycles. ... 11111111- Update interval value is 255 REF clock cycles.

### 25.3.65 NAND DLL Low Status Register (IFC\_NAND\_DLL\_LOW\_STAT)

This register is used to store the status of the DLL for interface frequency upto 133 MHz

Address: 153\_0000h base + 130Ch offset = 153\_130Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DLL_STS_REF_LOCK				DLL_STS_SLV_LOCK								DLL_STS_REF_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	DLL_STS_REF_SEL												DLL_STS_SLV_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IFC\_NAND\_DLL\_LOW\_STAT field descriptions**

Field	Description
0 DLL_STS_REF_LOCK	DLL Reference Delay line lock status
1–3 -	This field is reserved. Reserved
4 DLL_STS_SLV_LOCK	DLL Slave Delay Chain lock status
5–11 -	This field is reserved. Reserved
12–19 DLL_STS_REF_SEL	Status of selected tap for reference delay line
20–23 -	This field is reserved. Reserved
24–31 DLL_STS_SLV_SEL	Status of selected tap for slave delay line

### 25.3.66 NOR Event and Error Status register (IFC\_NOR\_EVTER\_STAT)

As with NAND FCM, 1 KB memory-mapped region is allocated for the NOR-specific registers. This section describes these registers.

NOR event and error status register (NOR\_EVTER\_STAT) indicates the cause of an error or event corresponding to NOR flash.

The following figure shows the register fields. It is write-1-to-clear register.

## IFC memory map/register definition

Address: 153\_0000h base + 1400h offset = 153\_1400h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	OPC_NOR					WPER		STOER								
W	w1c					w1c		w1c								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### IFC\_NOR\_EVTER\_STAT field descriptions

Field	Description
0 OPC_NOR	<p>NOR Command Sequence Operation Complete Event Indication</p> <p><b>NOTE:</b> Not valid in case of read operations which do not require a command sequence to be performed</p> <ul style="list-style-type: none"> <li>0 NOR Command Sequence Operation not completed</li> <li>1 Indicates the completion of a Command Sequence performed on the NOR flash device. A command sequence is said to have completed once all the phases (as indicated in the NORCR[NUM_PHASE]) have been sent to the NOR flash device.</li> </ul>

Table continues on the next page...

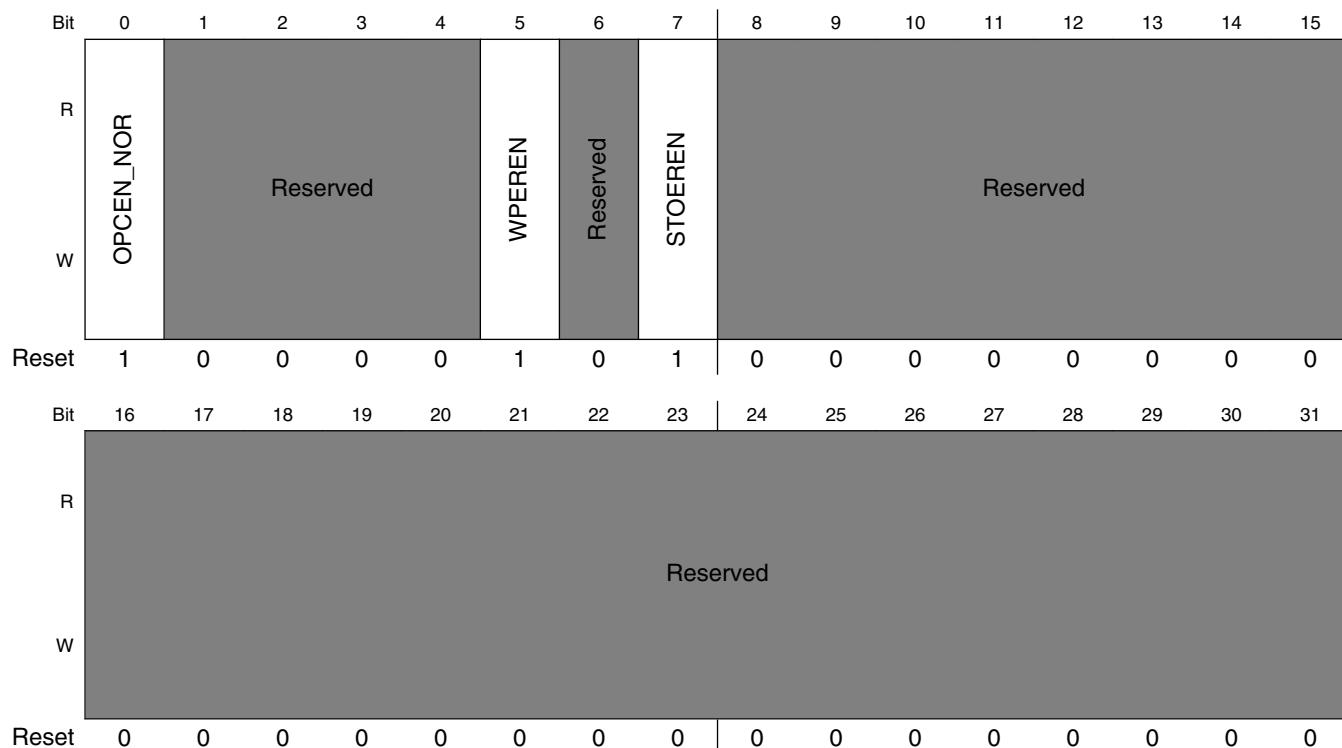
**IFC\_NOR\_EVTER\_STAT field descriptions (continued)**

Field	Description
1–4 -	This field is reserved.
5 WPER	Write Protect Error 0 No write protect error 1 A write is attempted to the memory bank which is write protected
6 -	This field is reserved.
7 STOER	Command sequence timeout error 0 No command sequence timeout from system side 1 Command sequence timeout from system side
8–31 -	This field is reserved.

**25.3.67 NOR Event and Error Enable register (IFC\_NOR\_EVTER\_EN)**

Event and Error Enable Register (NOR\_EVTER\_EN) is used to enable/disable the logging of event and error indication in the NOR\_EVTER\_STAT register.

Address: 153\_0000h base + 140Ch offset = 153\_140Ch



**IFC\_NOR\_EVTER\_EN field descriptions**

Field	Description
0 OPCEN_NOR	NOR Command Sequence operation complete event enable  0 OPC_NOR Event is not enabled 1 OPC_NOR Event is enabled
1–4 -	This field is reserved.
5 WPEREN	Write Protect Error Checking Enable  0 No write protect error checking 1 Write protect error checking is enabled
6 -	This field is reserved.
7 STOEREN	Command Sequence Timeout Error Enable  0 No command sequence timeout checking 1 Command sequence timeout error checking enable
8–31 -	This field is reserved.

### 25.3.68 NOR Event and Error Interrupt enable register (IFC\_NOR\_EVTER\_INTR\_EN)

Event and Error Interrupt Enable Register (NOR\_EVTER\_INTR\_EN) is used to enable/disable the generation of interrupt signals corresponding to error and event reporting. Software must clear pending events and errors in NOR\_EVTER\_STAT register before enabling the interrupts.

Address: 153\_0000h base + 1418h offset = 153\_1418h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	OPCIREN_NOR					WPERIREN		STOERIREN								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IFC\_NOR\_EVTER\_INTR\_EN field descriptions

Field	Description
0 OPCIREN_NOR	NOR command sequence operation complete event Interrupt enable 0 Event Interrupt is not enabled 1 Event Interrupt is enabled
1–4 -	This field is reserved.
5 WPERIREN	Write protect error interrupt enable 0 Write protect error interrupt disable 1 Write protect error interrupt is enabled

Table continues on the next page...

**IFC\_NOR\_EVTER\_INTR\_EN field descriptions (continued)**

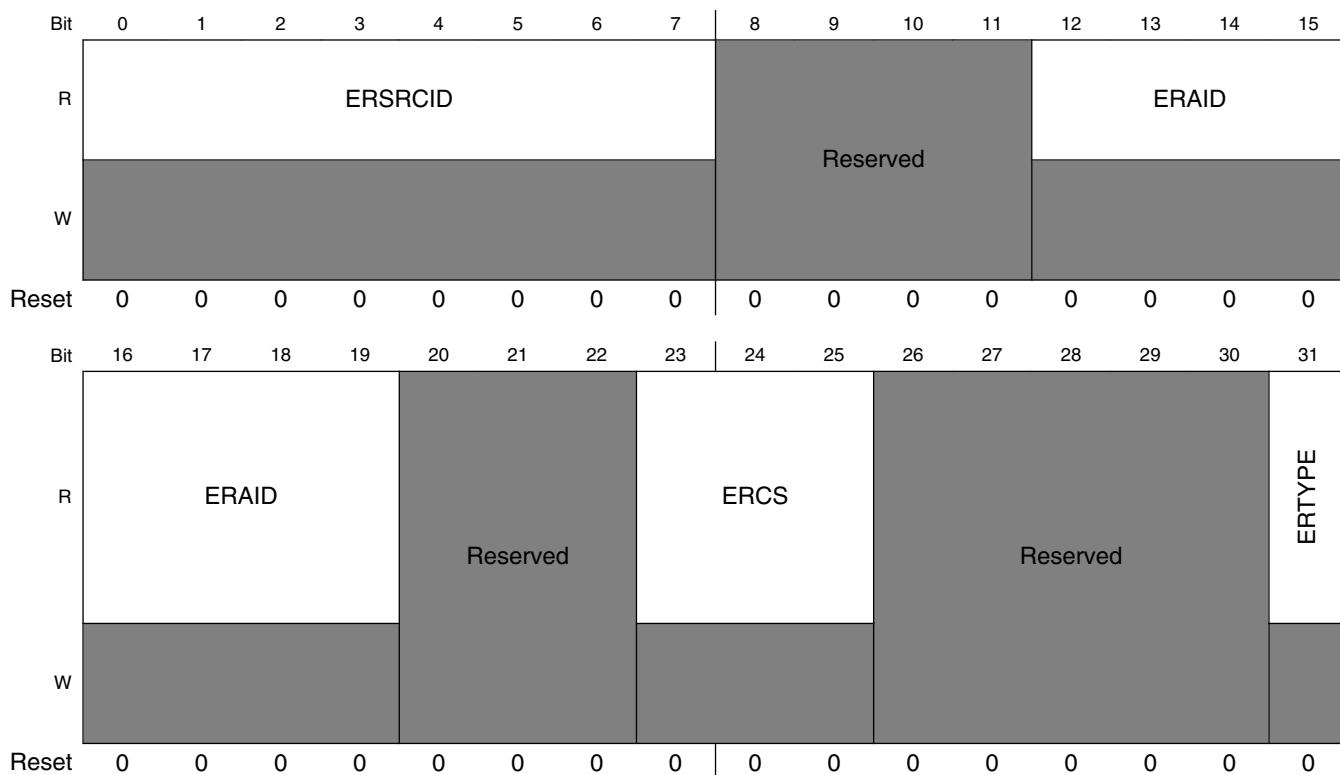
Field	Description
6 -	This field is reserved.
7 STOERIREN	NOR command sequence timeout error interrupt enable 0 Command Sequence Timeout error interrupt disable 1 Command Sequence Timeout error interrupt is enabled
8–31 -	This field is reserved.

### 25.3.69 NOR Transfer Error Attributes register 0 (IFC\_NOR\_ERATTR0)

These registers store the attribute corresponding to the first transaction on which error occurred.

Transfer error attribute register 0 (NOR\_ERATTR0) is used to register the transaction attributes corresponding to the first error.

Address: 153\_0000h base + 1424h offset = 153\_1424h



**IFC\_NOR\_ERATTR0 field descriptions**

Field	Description
0–7 ERSRCID	SRCID corresponding to error transaction
8–11 -	This field is reserved.
12–19 ERAID	ID of the error transaction. This field is valid only for write protect error.
20–22 -	This field is reserved.
23–25 ERCS	Chip-select corresponding to NOR error  000 Bank0 001 Bank1 010 Bank2 ... 111 Reserved
26–30 -	This field is reserved.
31 ERTYPE	Type of the transaction  0 Write 1 Read

**25.3.70 NOR Transfer Error Attribute register 1 (IFC\_NOR\_ERATTR1)**

Transfer error attribute register (NOR\_ERATTR1) is used to register the transaction address corresponding to the first error.

Address: 153\_0000h base + 1428h offset = 153\_1428h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ERADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_NOR\_ERATTR1 field descriptions**

Field	Description
0–31 ERADDR	In the case of a write protect error, this field reflects the address corresponding to which a write cycle was received from the system side.  In the case of a sequence timeout error, this field reflects the last address received from the system side

### 25.3.71 NOR Transfer Error Attribute register 2 (IFC\_NOR\_ERATTR2)

Transfer error attribute register (NOR\_ERATTR2) is used to register the transaction SRCID corresponding to the first error.

Address: 153\_0000h base + 142Ch offset = 153\_142Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								ER_NUM_PHASE_EXP				Reserved				ER_NUM_PHASE_PER				Reserved											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### IFC\_NOR\_ERATTR2 field descriptions

Field	Description
0–11 -	This field is reserved.
12–15 ER_NUM_PHASE_EXP	Number of phase expected in command sequence which timed-out by system side.  <b>NOTE:</b> This attribute is valid only for sequence timeout error.  0000 1 Phase 0001 2 Phase ... 1111 16 Phases
16–19 -	This field is reserved.
20–23 ER_NUM_PHASE_PER	Actual no. of command sequence phases performed on NOR flash before timeout occurred  <b>NOTE:</b> This attribute is valid only for sequence timeout error  0000 0 Phase 0001 1 Phase ... 1111 15 Phases
24–31 -	This field is reserved.

### 25.3.72 NOR Control register (IFC\_NORCR)

Address: 153\_0000h base + 1440h offset = 153\_1440h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				NUM_PHASE				Reserved				STOCNT				Reserved								Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IFC\_NORCR field descriptions**

Field	Description																																
0–3 -	This field is reserved.																																
4–7 NUM_PHASE	<p>No. of Address/Data phase on device for which chip-select has to be asserted:</p> <p>This field is used for the NOR devices for which CS has to be asserted for multiple address data cycles. It can be used for command based NOR, where CS remains asserted during unlock cycles and actual address/data phase.</p> <p>For example, if the total number of phases (that is, command and address/data) to be sent corresponding to a particular operation (such as program, block erase, and chip erase) is 5, then this field should be programmed to 0100. This would indicate to the NOR FCM that the CE needs to be asserted for five consecutive commands sent from the system side (there by for a complete operation)</p> <table> <tbody> <tr><td>0000</td><td>1 phase</td></tr> <tr><td>0001</td><td>2 phase</td></tr> <tr><td>0010</td><td>3 phase</td></tr> <tr><td>0011</td><td>4 phase</td></tr> <tr><td>0100</td><td>5 phase</td></tr> <tr><td>0101</td><td>6 phase</td></tr> <tr><td>0110</td><td>7 phase</td></tr> <tr><td>0111</td><td>8 phase</td></tr> <tr><td>1000</td><td>9 phase</td></tr> <tr><td>1001</td><td>10 phase</td></tr> <tr><td>1010</td><td>11 phase</td></tr> <tr><td>1011</td><td>12 phase</td></tr> <tr><td>1100</td><td>13 phase</td></tr> <tr><td>1101</td><td>14 phase</td></tr> <tr><td>1110</td><td>15 phase</td></tr> <tr><td>1111</td><td>16 phase</td></tr> </tbody> </table>	0000	1 phase	0001	2 phase	0010	3 phase	0011	4 phase	0100	5 phase	0101	6 phase	0110	7 phase	0111	8 phase	1000	9 phase	1001	10 phase	1010	11 phase	1011	12 phase	1100	13 phase	1101	14 phase	1110	15 phase	1111	16 phase
0000	1 phase																																
0001	2 phase																																
0010	3 phase																																
0011	4 phase																																
0100	5 phase																																
0101	6 phase																																
0110	7 phase																																
0111	8 phase																																
1000	9 phase																																
1001	10 phase																																
1010	11 phase																																
1011	12 phase																																
1100	13 phase																																
1101	14 phase																																
1110	15 phase																																
1111	16 phase																																
8–11 -	This field is reserved.																																
12–15 STOCNT	<p>Sequence Timeout Count</p> <p><b>NOTE:</b> This counter is used for timeout on sequential transactions on command based NOR.</p> <table> <tbody> <tr><td>0000</td><td>256 cycles of IFC module input clock</td></tr> <tr><td>0001</td><td>512 cycles of IFC module input clock</td></tr> <tr><td>0010</td><td>1024 cycles of IFC module input clock</td></tr> <tr><td>0011</td><td>2048 cycles of IFC module input clock</td></tr> <tr><td>0100</td><td>4096 cycles of IFC module input clock</td></tr> <tr><td>0101</td><td>8192 cycles of IFC module input clock</td></tr> <tr><td>0110</td><td>16,384 cycles of IFC module input clock</td></tr> <tr><td>0111</td><td>32,768 cycles of IFC module input clock</td></tr> <tr><td>1000</td><td>65,536 cycles of IFC module input clock</td></tr> <tr><td>1001</td><td>131,072 cycles of IFC module input clock</td></tr> <tr><td>1010</td><td>262,144 cycles of IFC module input clock</td></tr> <tr><td>1011</td><td>524,288 cycles of IFC module input clock</td></tr> <tr><td>1100</td><td>1,048,576 cycles of IFC module input clock</td></tr> <tr><td>1101</td><td>2,097,152 cycles of IFC module input clock</td></tr> </tbody> </table>	0000	256 cycles of IFC module input clock	0001	512 cycles of IFC module input clock	0010	1024 cycles of IFC module input clock	0011	2048 cycles of IFC module input clock	0100	4096 cycles of IFC module input clock	0101	8192 cycles of IFC module input clock	0110	16,384 cycles of IFC module input clock	0111	32,768 cycles of IFC module input clock	1000	65,536 cycles of IFC module input clock	1001	131,072 cycles of IFC module input clock	1010	262,144 cycles of IFC module input clock	1011	524,288 cycles of IFC module input clock	1100	1,048,576 cycles of IFC module input clock	1101	2,097,152 cycles of IFC module input clock				
0000	256 cycles of IFC module input clock																																
0001	512 cycles of IFC module input clock																																
0010	1024 cycles of IFC module input clock																																
0011	2048 cycles of IFC module input clock																																
0100	4096 cycles of IFC module input clock																																
0101	8192 cycles of IFC module input clock																																
0110	16,384 cycles of IFC module input clock																																
0111	32,768 cycles of IFC module input clock																																
1000	65,536 cycles of IFC module input clock																																
1001	131,072 cycles of IFC module input clock																																
1010	262,144 cycles of IFC module input clock																																
1011	524,288 cycles of IFC module input clock																																
1100	1,048,576 cycles of IFC module input clock																																
1101	2,097,152 cycles of IFC module input clock																																

*Table continues on the next page...*

**IFC\_NORCR field descriptions (continued)**

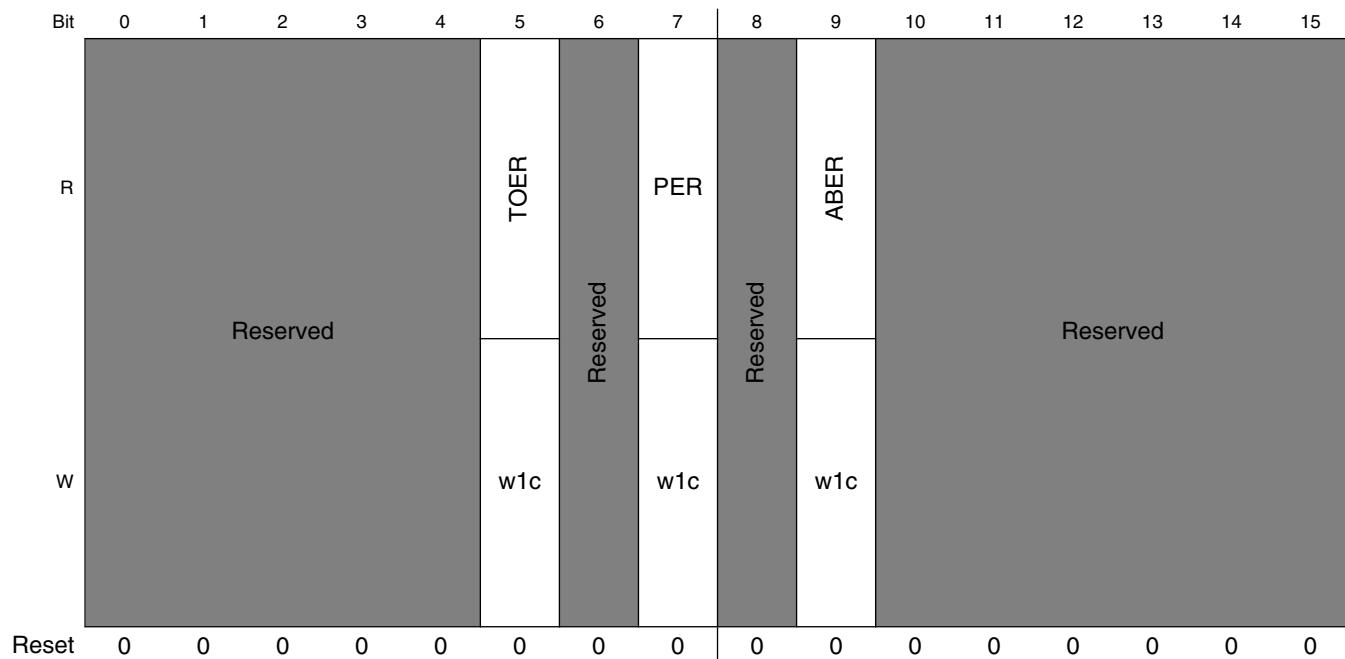
Field	Description
	1110 4,194,304 cycles of IFC module input clock 1111 8,388,608 cycles of IFC module input clock
16–31 -	This field is reserved.

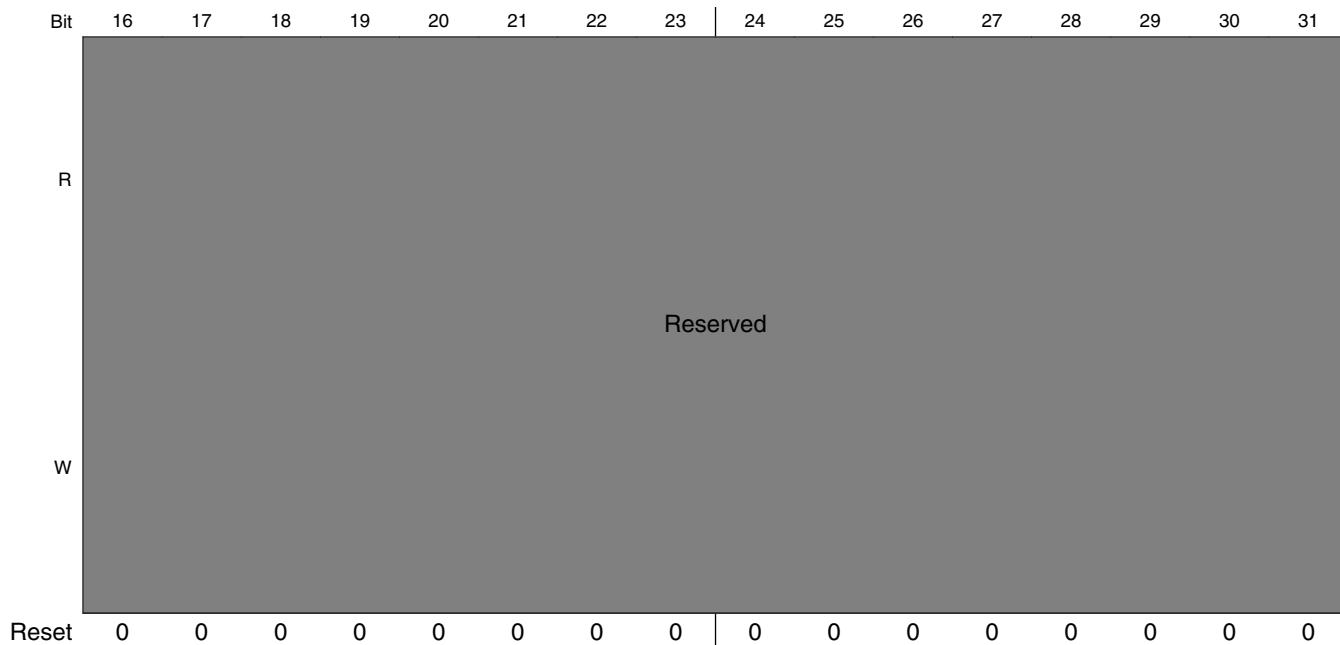
### 25.3.73 GPCM Event and Error Status register (IFC\_GPCM\_EVTER\_STAT)

A separate 1 KB memory-mapped region is allocated for the GPCM specific registers.

GPCM Event and Error Status Register (GPCM\_EVTER\_STAT) indicates the cause of an error or event corresponding to GPCM flash devices.

Address: 153\_0000h base + 1800h offset = 153\_1800h



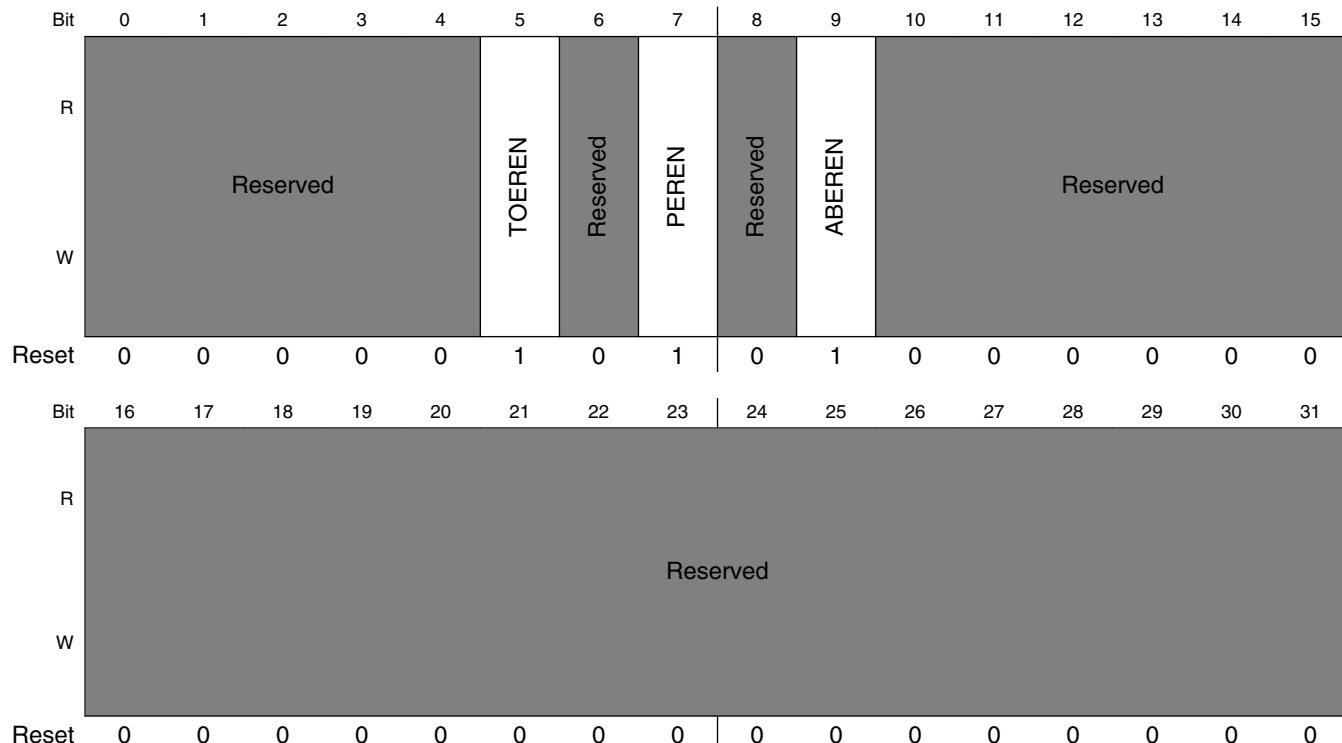
**IFC\_GPCM\_EVTER\_STAT field descriptions**

Field	Description
0–4 -	This field is reserved.
5 TOER	Timeout Error <b>NOTE:</b> Timeout for Normal GPCM mode will only be observed if IFCTA_B is configured as acknowledgement signal for transaction access completion, that is, by setting 1 in CSOR $n$ [WGETA] and CSOR $n$ [RGETA]. Timeout error will occur if IFC is waiting for the acknowledgement signal IFCTA_B to come and timeout counter value specified in CSOR[GPTO] counter has expired 0 No Timeout Error 1 Timeout observed for read/write transaction.
6 -	This field is reserved.
7 PER	Parity Error 0 No Parity Error 1 Parity Error for read/write transaction.
8 -	This field is reserved.
9 ABER	Abort Error. Abort for Normal GPCM mode will only be observed if access is terminated by IFCTA_B when CSOR $n$ [RGETA] is programmed to 0. It is valid for read transaction and no error is reported for write transaction. 0 No abort error 1 Abort happened for the current read transaction
10–31 -	This field is reserved.

### 25.3.74 GPCM Event and Error Enable register (IFC\_GPCM\_EVTER\_EN)

Event and Error Enable Register (GPCM\_EVTER\_EN) is used to enable/disable the logging of event and error indication in the GPCM\_EVTER\_STAT register.

Address: 153\_0000h base + 180Ch offset = 153\_180Ch



#### IFC\_GPCM\_EVTER\_EN field descriptions

Field	Description
0–4 -	This field is reserved.
5 TOEREN	Timeout error checking enable 0 No Timeout Error checking 1 Timeout Error checking is enabled
6 -	This field is reserved.
7 PEREN	Parity error checking enable. 0 No Parity Error Checking 1 Parity Error checking Enabled

Table continues on the next page...

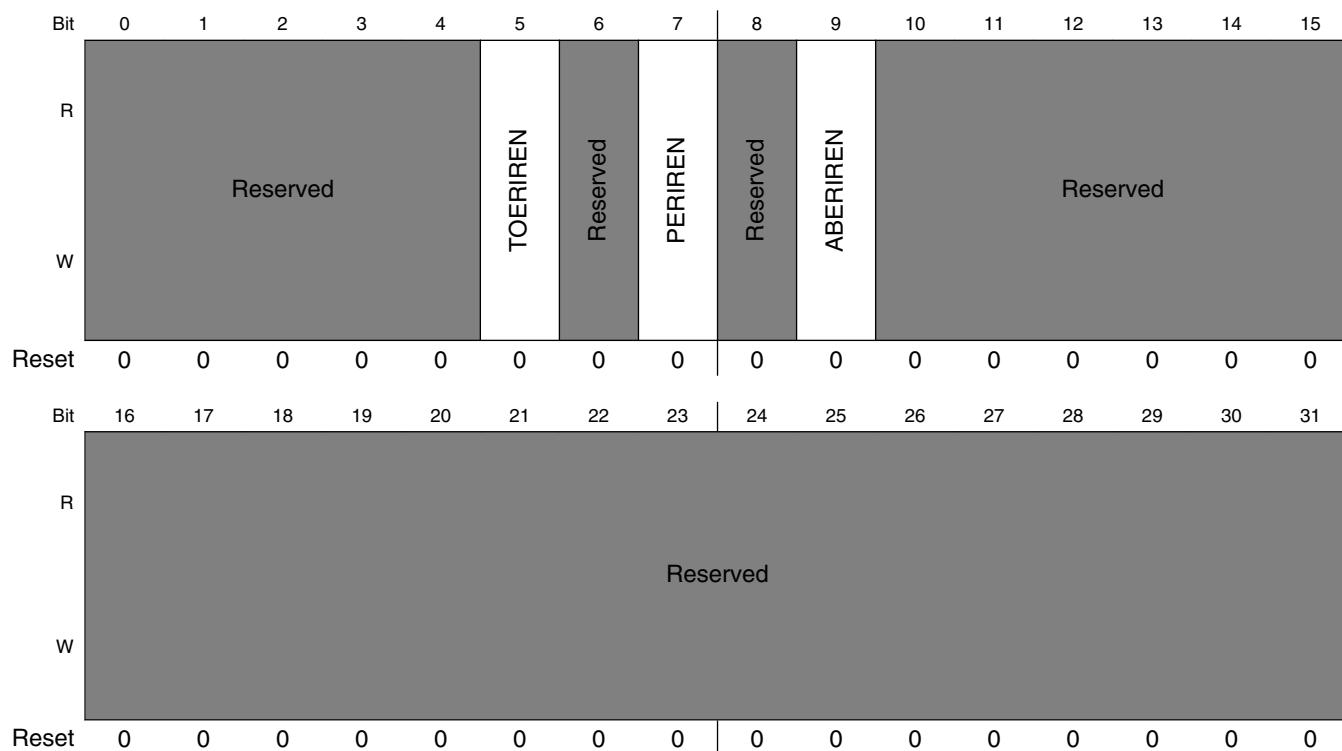
**IFC\_GPCM\_EVTER\_EN field descriptions (continued)**

Field	Description
8 -	This field is reserved.
9 ABEREN	Abort Error Checking Enable 0 No Abort Error checking 1 Abort Error checking is enabled
10–31 -	This field is reserved.

### 25.3.75 GPCM Event and Error Interrupt enable register (IFC\_GPCM\_EVTER\_INTR\_EN)

Event and Error Interrupt Enable Register (GPCM\_EVTER\_INTR\_EN) is used to enable/disable the generation of interrupt signals corresponding to error and event reporting. Software must clear pending events and errors in GPCM\_EVTER\_STAT register before enabling the interrupts.

Address: 153\_0000h base + 1818h offset = 153\_1818h



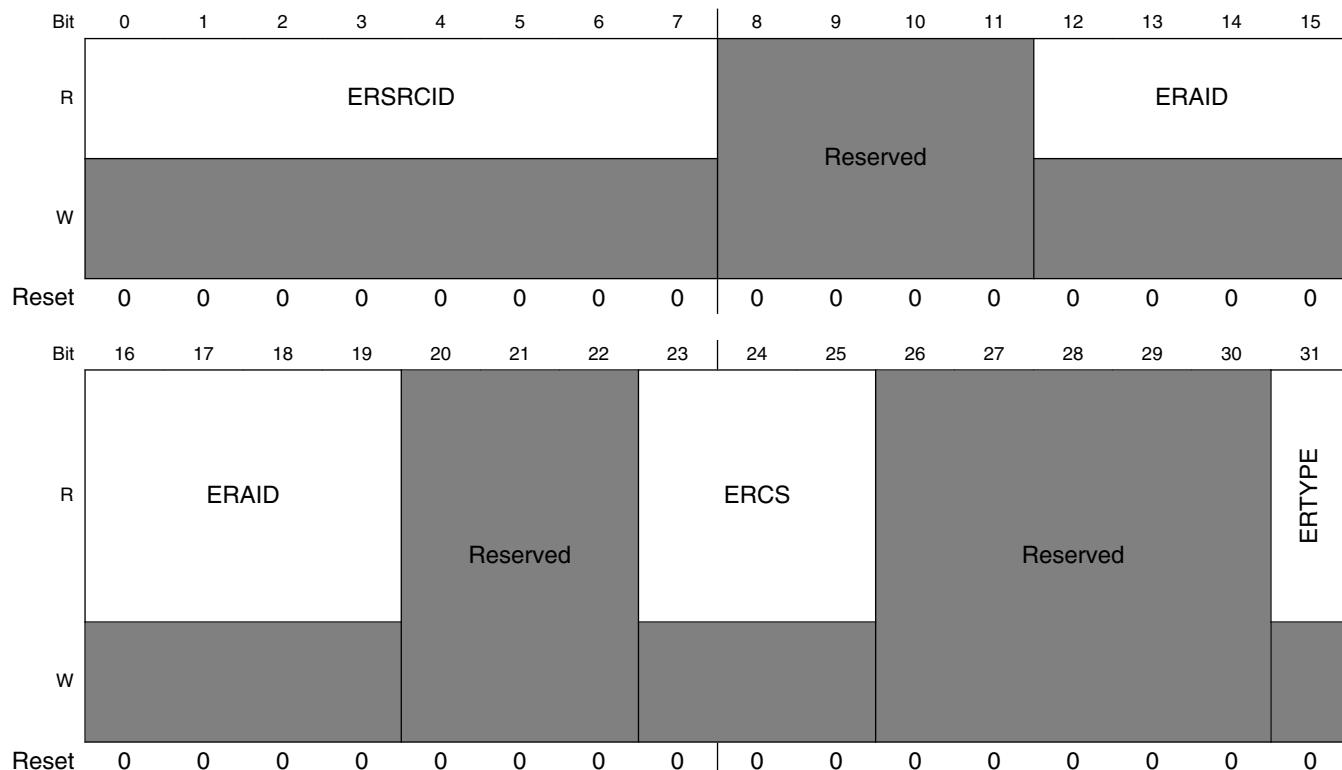
**IFC\_GPCM\_EVTER\_INTR\_EN field descriptions**

Field	Description
0–4 -	This field is reserved.
5 TOERIREN	Tmeout Error Interrupt Enable 0 Timeout Error interrupt disable 1 Timeout Error interrupt is enabled
6 -	This field is reserved.
7 PERIREN	Parity Error Interrupt Enable 0 Parity Error interrupt disable 1 Parity Error interrupt is enabled
8 -	This field is reserved.
9 ABERIREN	Abort Error Interrupt Enable 0 Abort Error interrupt disable 1 Abort Error interrupt is enabled
10–31 -	This field is reserved.

### 25.3.76 GPCM Transfer Error Attributes register 0 (IFC\_GPCM\_ERATTR0)

Transfer error attribute register 0 (GPCM\_ERATTR0) is used to register the transaction attributes corresponding to error occurred.

Address: 153\_0000h base + 1824h offset = 153\_1824h



**IFC\_GPCM\_ERATTR0 field descriptions**

Field	Description
0–7 ERSRCID	SRCID corresponding to error transaction
8–11 -	This field is reserved.
12–19 ERAID	ID of the error transaction
20–22 -	This field is reserved.
23–25 ERCS	ERCS 000 Bank0 001 Bank1

*Table continues on the next page...*

**IFC\_GPCM\_ERATTR0 field descriptions (continued)**

Field	Description
	010 Bank2 . . . 111 Reserved
26–30	This field is reserved.
31 ERTYPE	Type of the transaction 0 Write 1 Read

### 25.3.77 GPCM Transfer Error Attributes register 1 (IFC\_GPCM\_ERATTR1)

Transfer error attribute register (GPCM\_ERATTR1) is used to register the transaction address corresponding to error occurred.

Address: 153\_0000h base + 1828h offset = 153\_1828h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

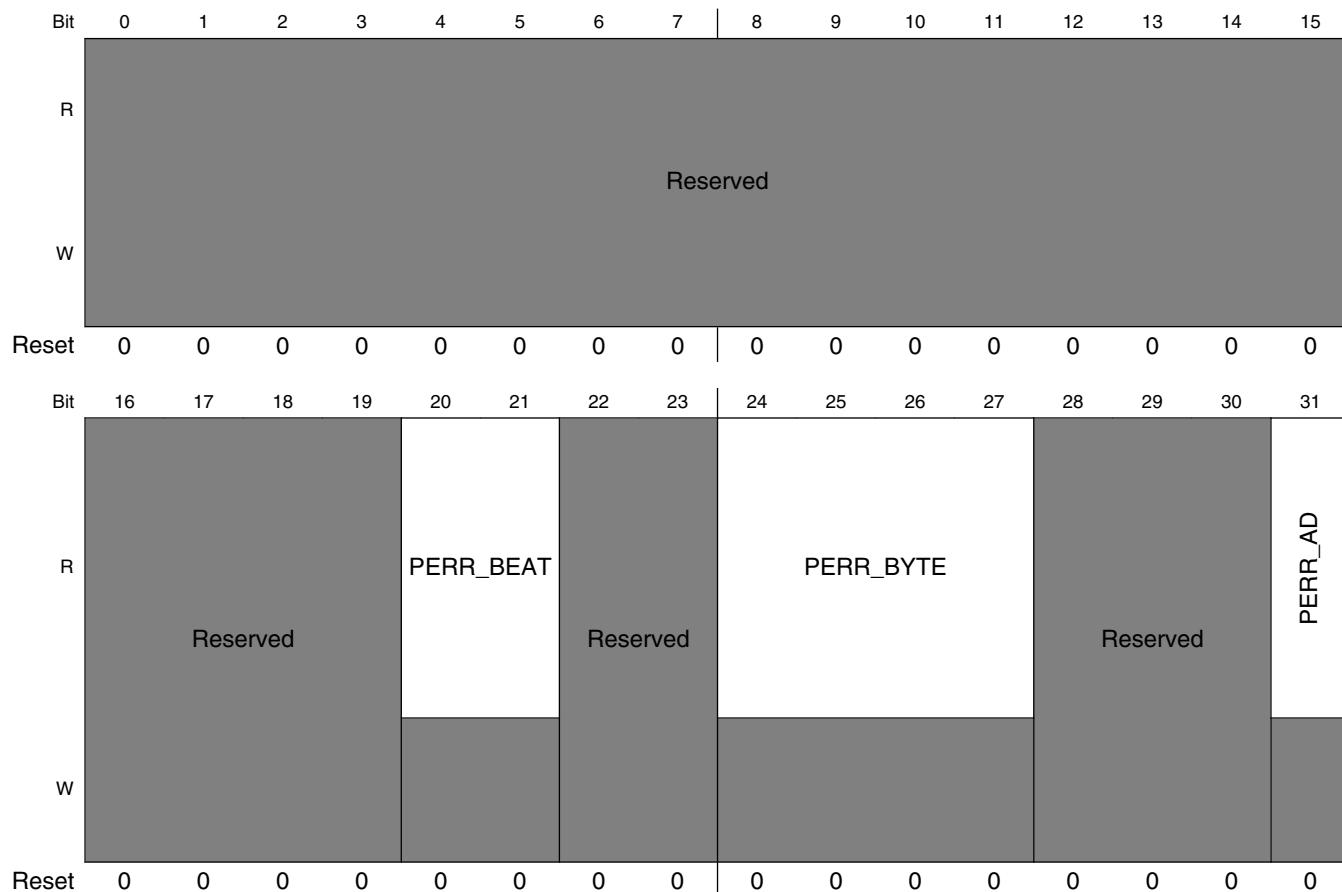
**IFC\_GPCM\_ERATTR1 field descriptions**

Field	Description
0–31 ERADDR	Address corresponding to error transaction

### 25.3.78 GPCM Transfer Error Attributes register 2 (IFC\_GPCM\_ERATTR2)

Transfer error attribute register (GPCM\_ERATTR2) is used to register the transaction attributes corresponding to error occurred.

Address: 153\_0000h base + 182Ch offset = 153\_182Ch



**IFC\_GPCM\_ERATTR2 field descriptions**

Field	Description
0–19 -	This field is reserved.
20–21 PERR_BEAT	This gives information on which beat of address/data parity error is observed. This has value from 0 to 3. For example, if port size is 16 bit and parity error is observed on second beat, then value 1 is reported in the status register. This field is valid for Generic ASIC mode.
22–23 -	This field is reserved.

*Table continues on the next page...*

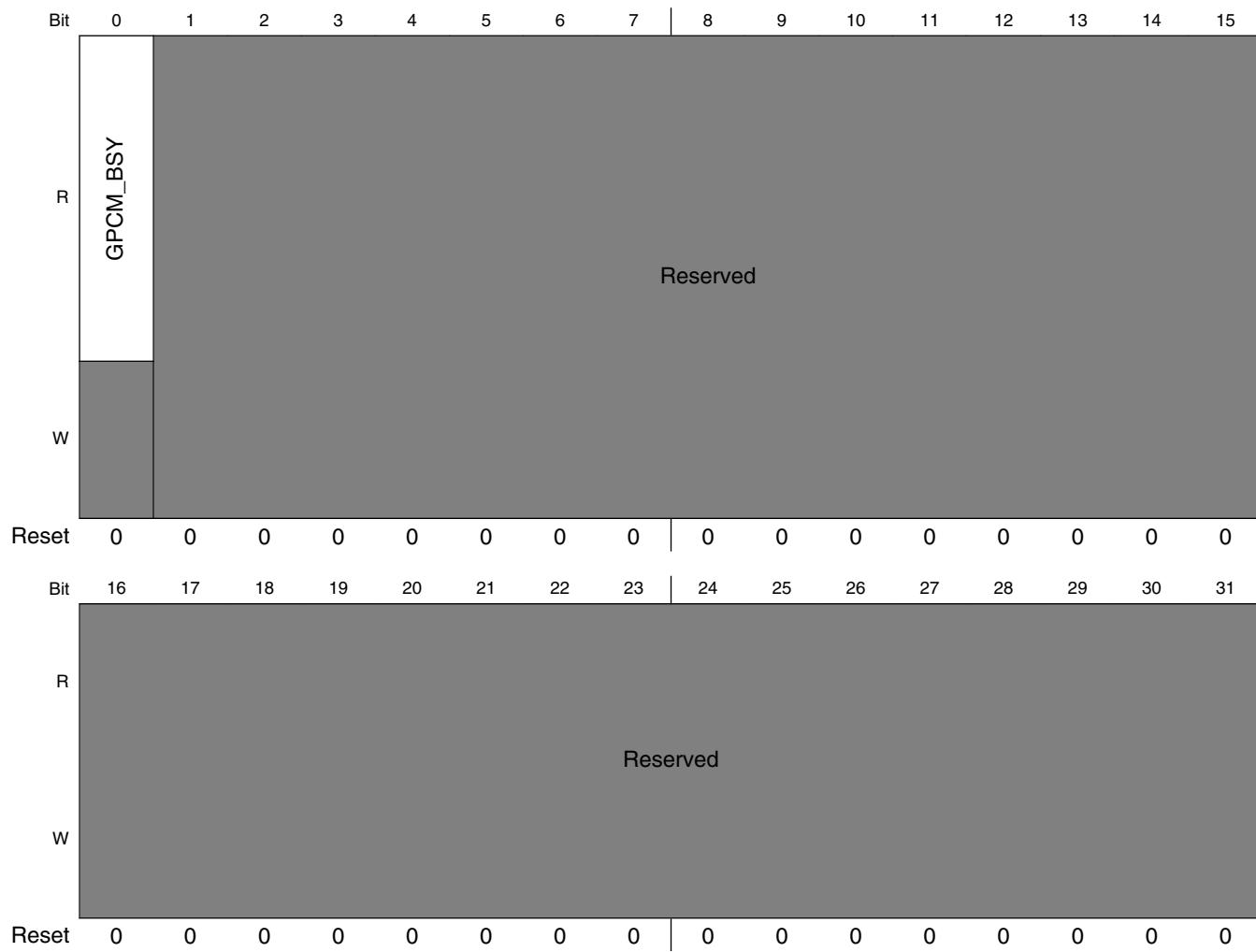
**IFC\_GPCM\_ERATTR2 field descriptions (continued)**

Field	Description
24–27 PERR_BYTE	Parity Error on byte. For GPCM there are four parity error status bit, one per byte. A bit is set for the byte that has parity error (bit 24 represents byte 0, the most significant byte lane). This field is valid for Normal GPCM mode.
28–30 -	This field is reserved.
31 PERR_AD	Parity Error reported in address or data phase. Address phase is only valid for Generic ASIC mode of GPCM.  0 Address phase 1 Data Phase

### 25.3.79 GPCM Status register (IFC\_GPCM\_STAT)

This register is used to reflect the busy status of the GPCM controller.

Address: 153\_0000h base + 1830h offset = 153\_1830h



**IFC\_GPCM\_STAT field descriptions**

Field	Description
0 GPCM_BSY	GPCM_BSY 0 No transaction being done by GPCM controller. 1 GPCM controller is busy with transactions.
1–31 -	This field is reserved.

## 25.4 IFC functional description

The IFC is used to interface with external NAND flash, asynchronous NOR flash, SRAM, EPROM, and generic ASIC devices.

To achieve this functionality, the logic is partitioned in independent flash control machines (FCMs) in the IFC to control the timings of NAND flash, NOR flash, and GPCM separately. Seven independent chip-selects are provided, but they all share the same pins; hence, only one memory can be accessed at a time based on the machine-select bits of the chip-select property register for that bank (CSPR $n$ [MSEL]). If a bank match occurs, the corresponding machine (NAND, NOR, or GPCM) takes ownership of the external signals that control the access and maintains control until the transaction ends.

This figure is a functional block diagram of the flash memory interface and its signal muxing.

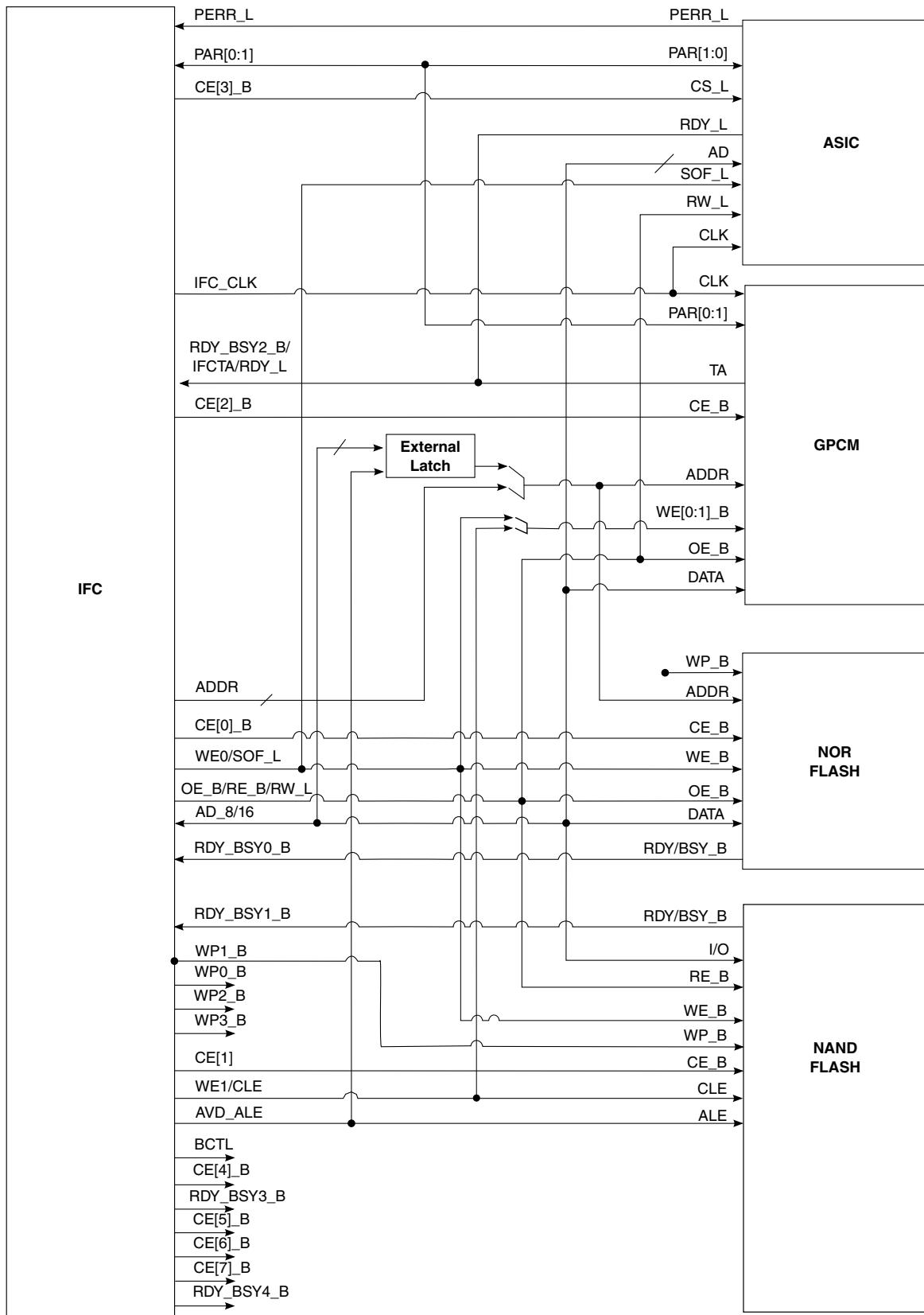


Figure 25-6. Flash memory interface and muxing (with NVDDR NAND)

The IFC supports the following types of flash control machines (FCMs):

- NOR FCM provides interfacing with NOR flash memories that have a 8-/16-bit-wide data bus. NOR FCM-controlled banks are used primarily for booting (direct memory-mapped) and code storage.
- The NAND FCM interfaces to NAND flash EEPROMs with 8- and 16-bit data buses. If NAND is chosen as the booting device, after reset, the NAND FCM can load boot code into SRAM buffer for execution. Following boot, the NAND FCM provides a flexible instruction sequencer that allows a user-defined command, address, and data transfer sequence of up to 15 steps to be executed against a memory-mapped buffer RAM. An advance ECC algorithm (BCH codes) is implemented to correct up to 4-/8-bit errors per sector of 512 bytes and 24/40 bit errors per 1KB sector.
- GPCM provides an interface to simple, synchronous, memory-mapped devices.

Each memory bank can be assigned to any of these types of machines through the machine-select bits of the chip-select property register for that chip-select (CSPRn[MSEL]).

## 25.4.1 General architecture

The basic architecture of the IFC allows bank selection through address decoding and address/data pin muxing.

### 25.4.1.1 Bank selection through address decoding

Banks can be selected via address decoding.

The defined base addresses are written to CSPRn[BA] and CSPRn\_EXT[BA\_EXT], while the corresponding address masks are written to AMASKn[AM]. Each time a local system access is requested, the internal transaction address is compared with each bank. Address decoding logic is explained in [Address Mask register \(IFC\\_AMASnK\)](#). If a match is found on a memory controller bank, the attributes defined in the CSPRn and CSORn for that bank are used to control the memory access. If a match is found in more than one bank, the lowest-numbered bank handles the memory access (that is, bank 0 has priority over bank 1).

### 25.4.1.2 Address/Data pin muxing for external address latch

There are various pin-muxing schemes available at the IFC interface.

The IFC provides muxing of the address and data bits on the same bus (AD), where the muxing is controlled by the AVD/ALE signal. There are two modes supported to supply, either address msbs (most significant bits) or lsbs (least significant bits) on AD bus.

Chip-select register field CSORn[ADM\_SHFT\_MODE] determines the mode of address-data pin muxing for the chip. The pin-muxing modes are described in [Mode 0 pin muxing \(CSORn\[ADM\\_SHFT\\_MODE\] = 0\)](#) and [Mode 1 pin muxing \(CSORn\[ADM\\_SHFT\\_MODE\] = 1\)](#).

#### 25.4.1.2.1 Mode 0 pin muxing (CSORn[ADM\_SHFT\_MODE] = 0)

In this mode of muxing, the IFC supplies the most significant bit (msb) of the address bits on the AD bus. Register field CSORn[ADM\_SHFT] controls the amount of address shift to align the msb of address with AD[0]. This shift can be utilized to align the msb of system address with the AD[0] signal during address phase (when AVD/ALE is asserted).

An application of this mode is an external latched-based system where the AVD/ALE signal is connected to the latch enable such that external latch latches the address msb coming through the AD bus during AVD/ALE assertion. This mode is used to connect conventional NOR devices (having dedicated address and data pins) as well as GPCM-based interfaces.

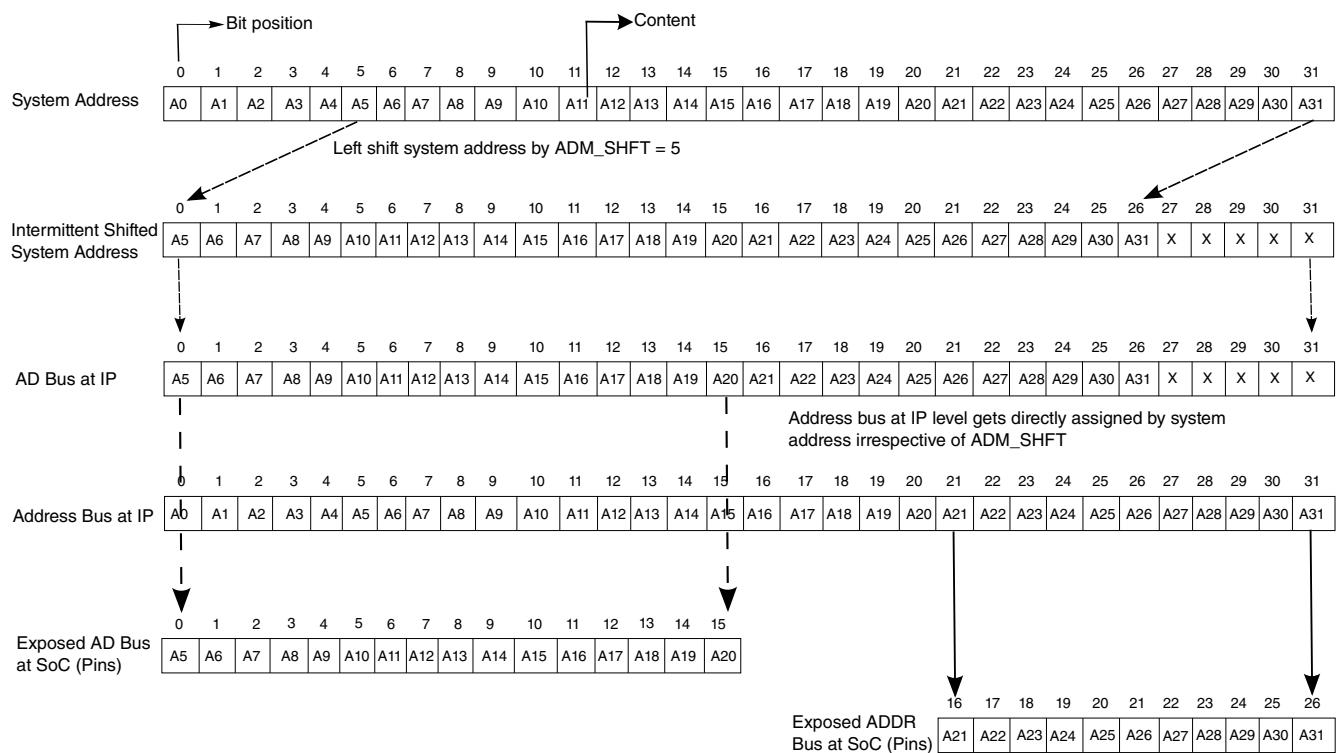
In this mode, system address is left shifted by CSORn[ADM\_SHFT] shift value and assigned to AD[0:31]. The address msb will be assigned to data bus msb (AD[0]). The shifted address can be latched by external latch at the falling edge of AVD/ALE.

The system address directly gets assigned to the IFC address bus (ADDR [0:31]) irrespective of the ADM\_SHFT value. The least significant bit (lsb) can be retrieved from the ADDR bus. ADDR[31] will carry the lsb of the system address.

To interface with a x16 NOR device, the address lsb must be left unconnected on the board.

For a chip that has a 16-bit AD bus and an 11-bit ADDR bus, which needs to be interfaced with a memory of 8-bit port width and 128 MB memory size (requiring a 27-bit address), the following figure shows how system address bits are placed during IFC address phase (AVD/ALE assertion) with a CSORn[ADM\_SHFT] value of 5.

## IFC functional description

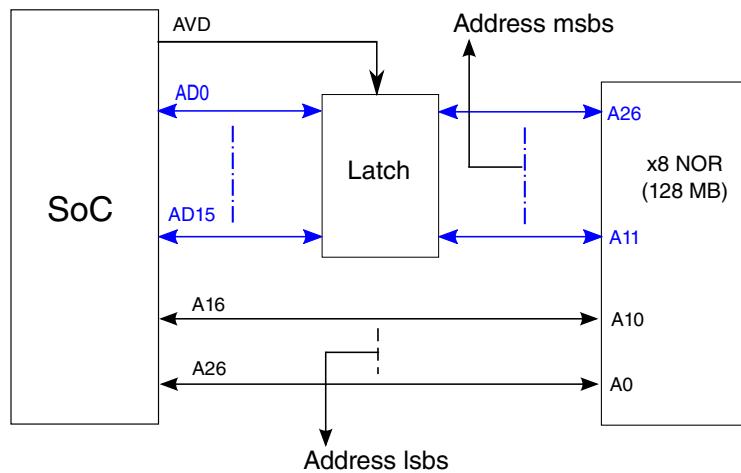


**Figure 25-7. System address assignment with CSORn[ADM\_SHFT]= 5 for ADM MODE 0**

These figures show a x8 and x16 memory connection for the configuration given above, where the shift value is 5.

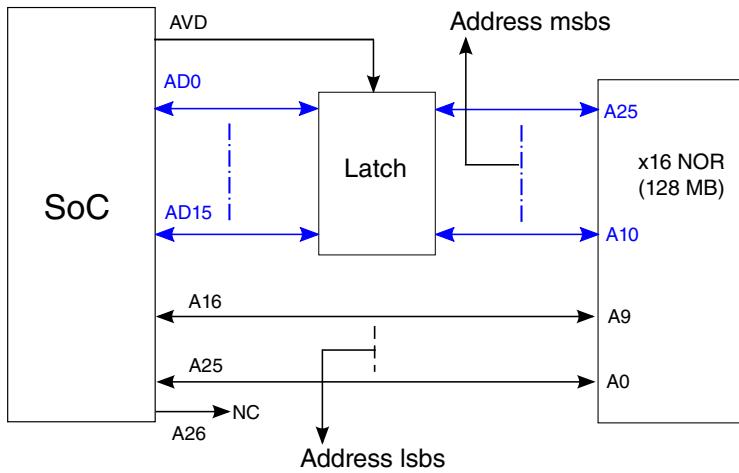
### NOTE

These figures are intended to be examples only and may show ADDR pins that are not supported.



**Figure 25-8. Connection of x8 NOR for ADM MODE 0**

For a x16 memory connection, since the memory expects a 26-bit word address, the address lsb is left unconnected at the board to supply the word address.



**Figure 25-9. Connection of x16 NOR for ADM MODE 0**

#### 25.4.1.2.2 Mode 1 pin muxing (CSORn[ADM\_SHFT\_MODE] = 1)

In this mode of muxing, the IFC supplies the address least significant bit (lsb) on the AD bus. CSORn[ADM\_SHFT] controls the address shift to align the system address lsb with AD[0]. In this mode, the ADDR bus carries the address most significant bit (msb). This mode is used to interface address data multiplexed (ADM) NOR devices (internal latch-based).

Connection in this mode should be treated as a special case since the board connectivity is in decrementing (reverse) bit order, as shown in following examples.

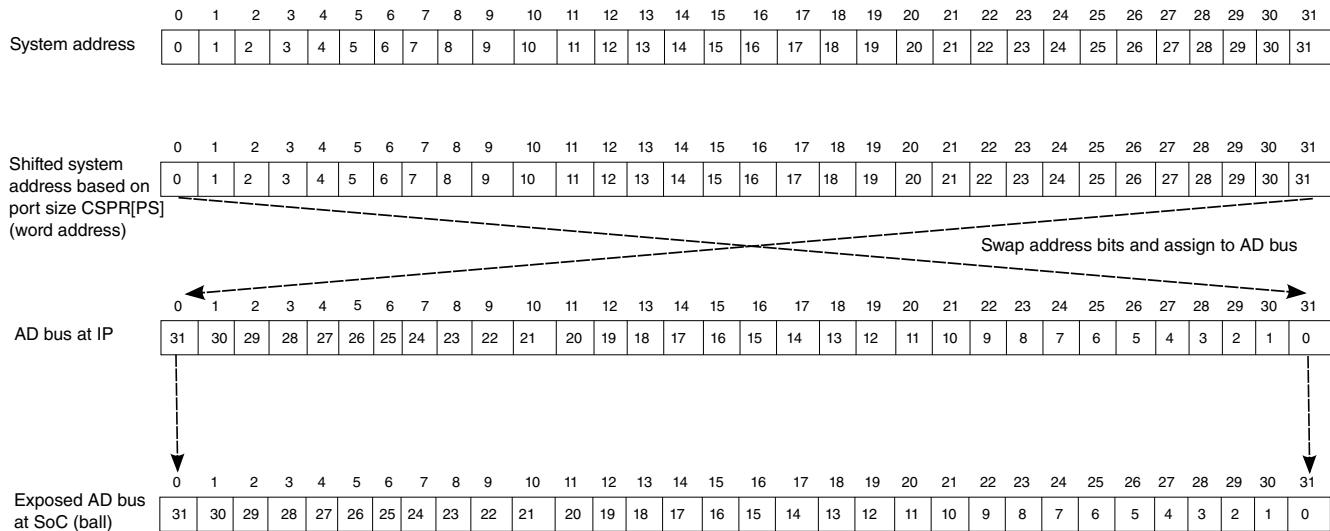
This is done to simplify the board connection for the ADM NOR as these devices expect the lsb to get latched at their internal latch with the assertion of AVD. In this mode, the system address remains the same and the IFC internally rearranges (based on the port size) the data bus in order to maintain the same data image for both the conventional NOR (external latched-based) as well as the ADM NOR (internal latch-based). This data rearrangement is performed internally within the IFC during both the read and write phases because the board connection for ADM NOR is opposite of the conventional (non-ADM) NOR devices. This mode is valid only for NOR mode of operation and not for GPCM.

Unlike in CSORn[ADM\_SHFT\_MODE] = 0, there is no need to leave the ADDR lsb unconnected on the board to supply word address to memory for x16- and x32-bit memories. The reason is that the lsb are now carried by the AD bus, which cannot be left unconnected since they carry the data during the data phase. Therefore, the logic is implemented in such a way that the IFC does a necessary address shift internally to supply word addresses to memory.

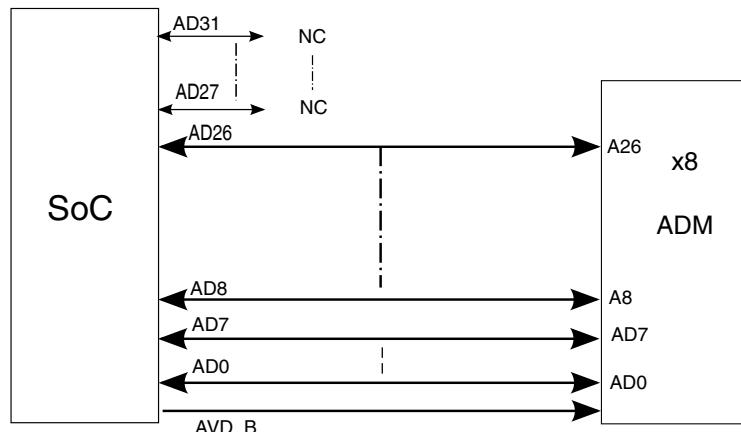
## IFC functional description

**Example 1:** The chip has exposed a 32-bit AD bus and x8 ADM NOR of 128 MB is connected through it.

In this example, ADM\_SHFT does not have any role as the chip has not exposed the ADDR bus and all address bits are available through the AD bus.



**Figure 25-10. Example 1 - System address assignment to the AD bus for ADM MODE 1**

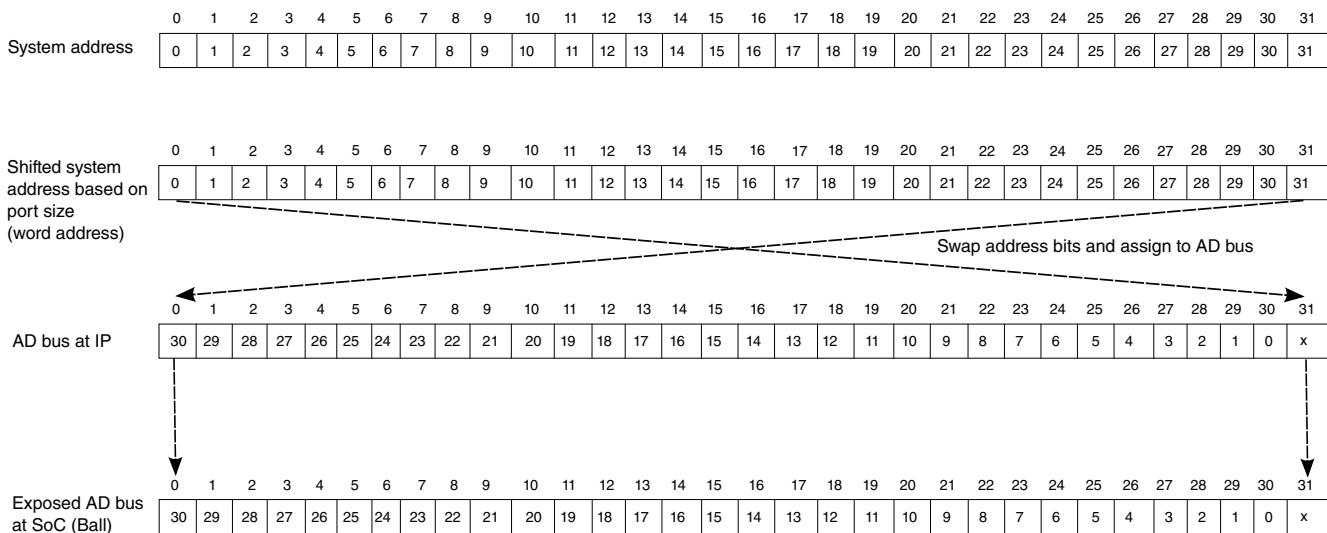


**Figure 25-11. Example 1 - Connection of x8 ADM NOR for ADM MODE 1**

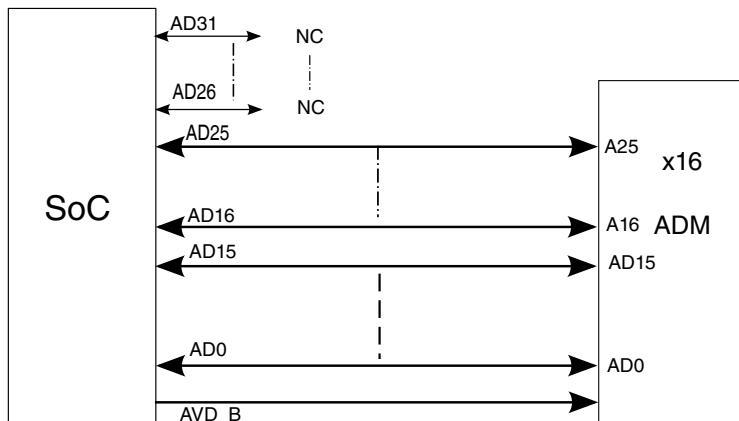
### NOTE

The board connections are in decrementing (reverse) bit order for the ADM NOR only.

**Example 2:** The chip has exposed a 32-bit AD bus and x16 ADM NOR of 128 MB is connected through it.



**Figure 25-12. Example 2 - System address assignment to the AD bus for ADM MODE 1**



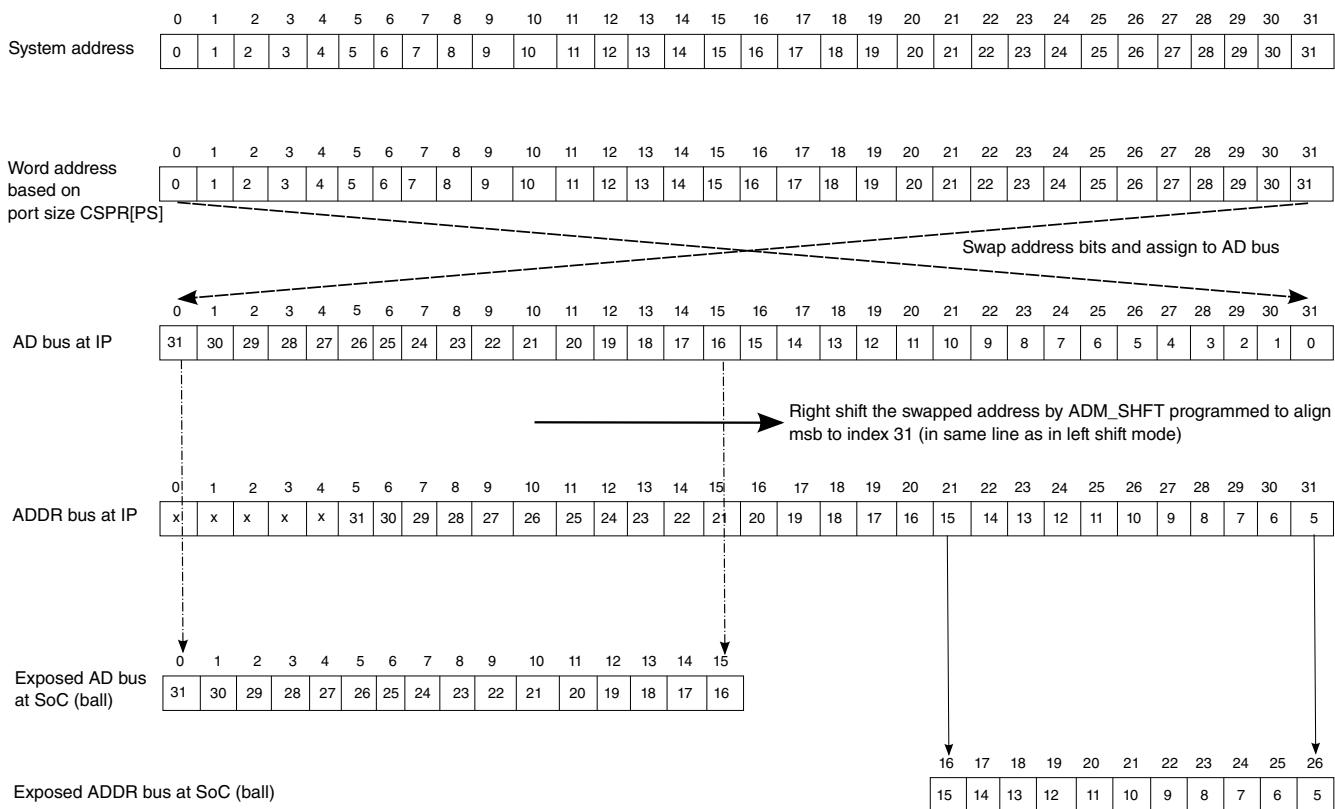
**Figure 25-13. Example 2 - Connection of x16 ADM NOR for ADM MODE 1**

Based on the port size, the address is generated such that the proper word address goes to the memory, unlike conventional NOR (ADM MODE 0). The lsb cannot be left unconnected as they are carried by AD[0] and is also used in the data phase.

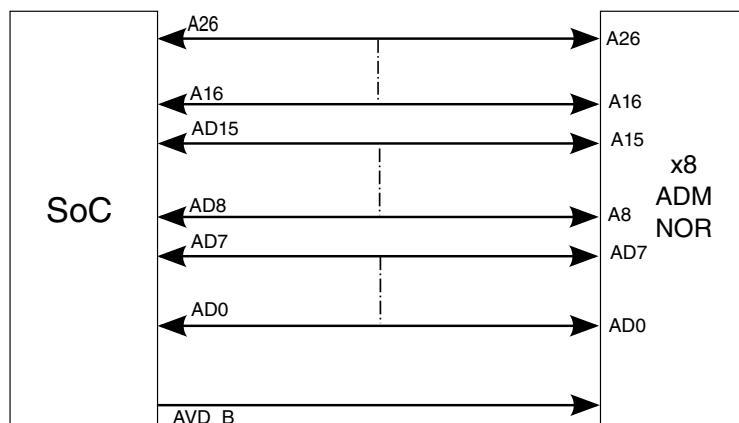
**Example 3:** The chip has exposed a 16-bit AD bus, 11-bit ADDR bus, and x8 ADM NOR of 128 MB is connected through it.

In this example, ADM\_SHFT=5 is required to align the system address msb with the right-most ADDR index as shown in the figure below.

## IFC functional description



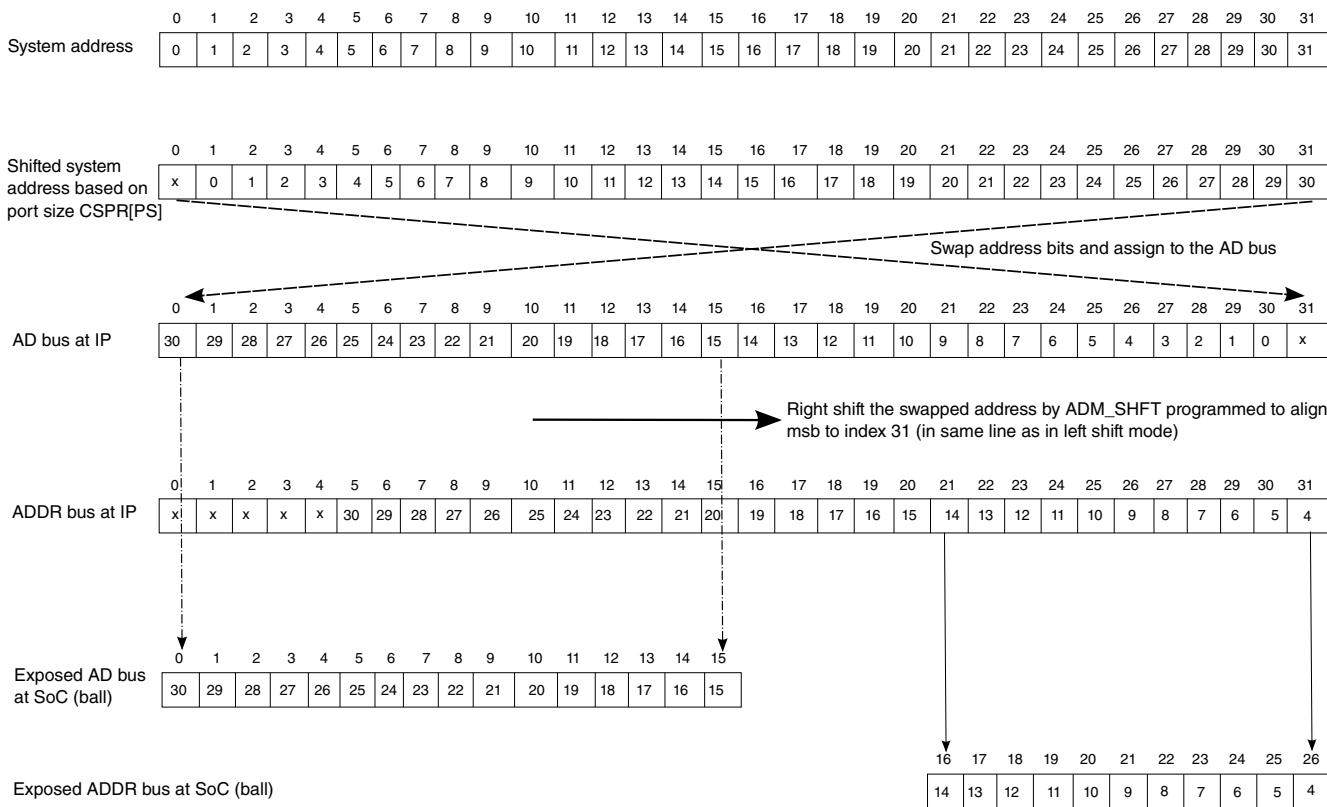
**Figure 25-14. Example 3 - System address assignment to the AD and ADDR bus for ADM MODE 1**



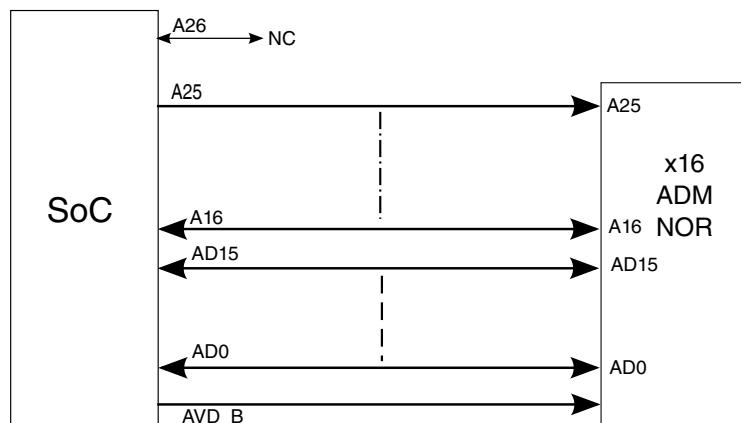
**Figure 25-15. Example 3 - Connection of x8 ADM NOR for ADM MODE 1**

**Example 4:** The chip has exposed a 16-bit AD bus, 11-bit ADDR bus, and x16 ADM NOR of 128 MB is connected through it.

In this example, ADM\_SHFT=5 is required to align the system address msb with the right-most ADDR index as shown below.



**Figure 25-16. Example 4 - System address assignment to AD and ADDR bus for ADM MODE 1**



**Figure 25-17. Example 4 - Connection of x16 ADM NOR for ADM MODE 1**

### NOTE

Timing of AVD and AVD\_B is same. Both are generated by the same logic except they are opposite in polarity. If the chip is only exposing AVD and not exposing AVD\_B, then it can be generated by using an on-board inverter and can be used for interfacing with ADM NOR.

## 25.4.2 Programming model for flash interface timing

The IFC follows a counter-based approach to generate the required timings on the flash side.

Programming registers (FTIM $n$ ) are provided that correspond to the various flash timing parameters. The values to be programmed in these registers are in terms of the number of IFC input clock-cycles derived by the following formula:

$$\text{Timing counter value} = \lceil \{\text{Flash Timing Parameter}(ns) \times \text{IFC module input clock frequency (MHz)}\} + 1000 - 1 \rceil / 1000$$

In this approach, counters running at the IFC module input clock generate the appropriate values for the flash interface signals depending on the value programmed in the Timing Registers (FTIM0-FTIM3). These timing registers must be programmed per the values specified in the flash device datasheet mapped to the waveforms shown in [NAND asynchronous mode timings](#), [Unmuxed \(parallel\) asynchronous NOR read timings](#), and [Simple asynchronous NOR write timings](#).

## 25.4.3 Booting methods

Booting can be performed from NAND or NOR flash on the IFC.

- NOR is directly memory-mapped, so booting from NOR can be done as simple read operations without any special arrangements. Default timing parameters are loaded with the assertion of rcw\_load/boot\_load, details of this requirement is explained in [NOR boot](#).
- NAND is not directly memory-mapped, so the boot code is loaded first into an SRAM buffer from where the code can be executed. See [NAND flash control machine](#) for more information about booting from NAND.

The flash timing registers ([Flash Timing register 0 for Chip Select n - NAND flash asyncNVDDR mode \(IFC\\_FTIM0\\_CSn\\_NAND\)](#) through [Flash Timing register 3 for Chip Select n - NAND Flash Mode \(IFC\\_FTIM3\\_CSn\\_NAND\)](#)) that control chip-select 0 have default values corresponding to the slowest-mode devices to guarantee boot.

Reset initialization is performed by the reset controller and the default configuration is determined by cfg\_rcw\_src and/or RCW settings.

## 25.4.4 Software reset handling

Software reset requires that software read and clear specific registers.

Software follows these steps as part of software reset handling:

1. Read all the event and error status registers.
2. Clear the NANDSEQ\_STRT register.
3. Clear the NAND\_AUTOBOOT\_TRGR register.
4. Apply soft reset.

After the soft reset has been applied, the hardware automatically clears all the event and error status registers. None of the other registers are reset. Therefore, the initial configuration done by the software is preserved.

### 25.4.4.1 System bus handling

Any pending transactions on the system bus are terminated with a valid response.

During software reset assertion, providing a good response that corresponds to the pending transaction on the IFC helps prevent a corresponding interrupt. The core expects an error interrupt whenever there is an error response from the system bus. Therefore, in the case of a software reset and an error response is provided, the core goes into a hang state because there is no corresponding interrupt. If there are pending read transactions, the garbage data is supplied back; if there is a pending write transaction, the data is not written in the memory.

The hardware does not allow the software to clear the soft reset bit until all pending transactions on the system bus have been gracefully terminated (with a good response). When there are no more transactions pending on the system bus, the hardware allows the software to clear the soft reset bit.

### 25.4.4.2 Flash interface handling

All the state machines (NAND, NOR, GPCM, and GASIC) are reset to the idle state.

All related FIFOs and registers are flushed, so the flash interface would be in the idle state.

## 25.4.5 Data buffer control (BCTL)

The IFC provides a data buffer control signal (BCTL) that can be disabled (remain high) by setting CSORn[BCTL\_D]. BCTL should be used to signify the write direction when high.

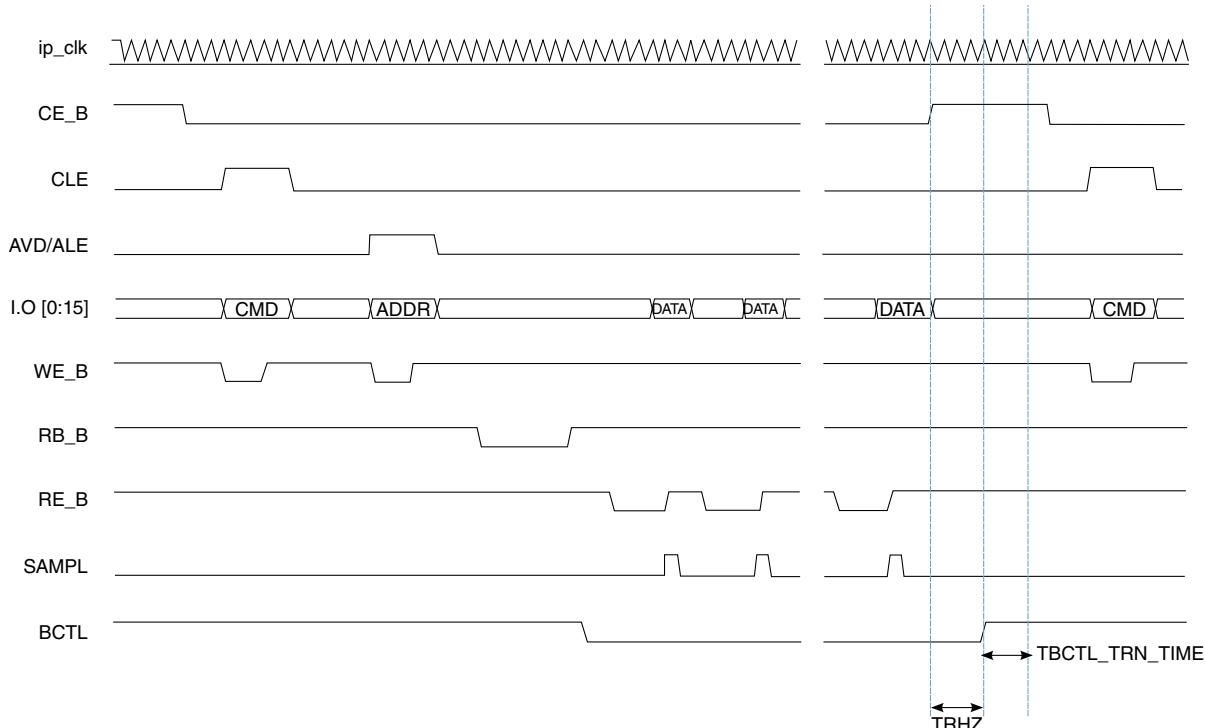
### NOTE

BCTL is a static value; it cannot be programmed on the fly.

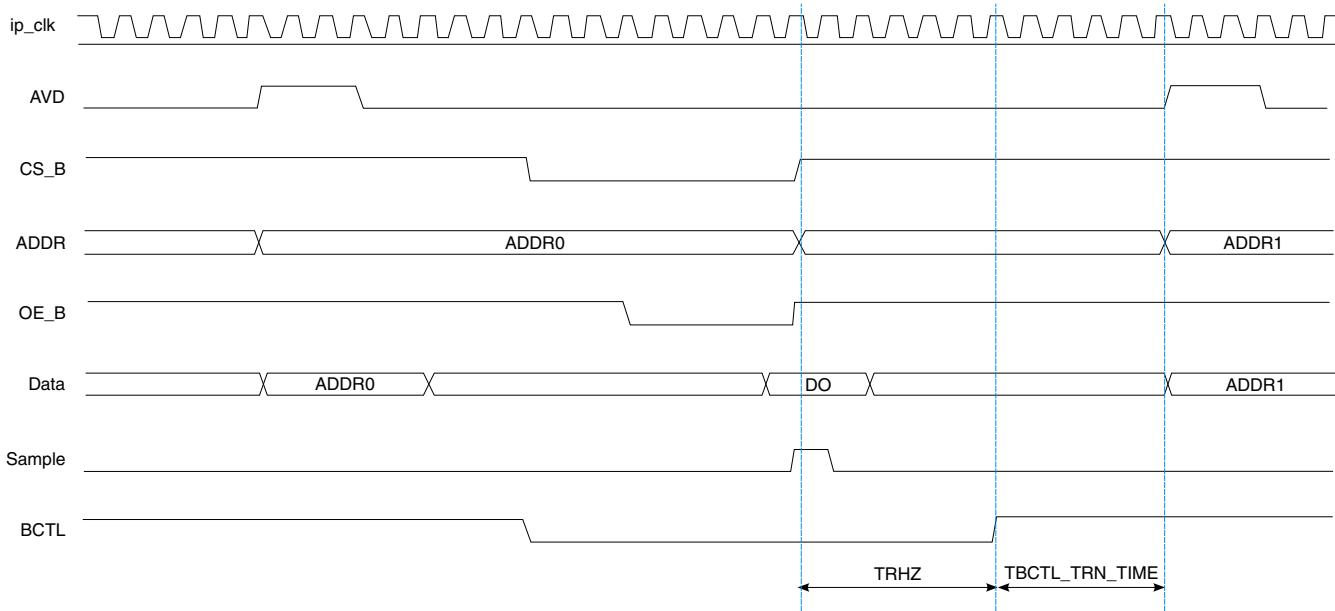
If the access is a write, BCTL remains high for the whole duration. However, if the access is a read, BCTL is negated (low) so that the memory device is able to drive the bus. Note that the default (reset and bus idle) value of BCTL is also high.

While accessing slow memories, the data driven by the flash is available on the bus after deassertion of read enable. To avoid bus contention, BCTL remains low (read mode) for the time defined by the CSORn[TRHZ] field.

Apart from CSORn[TRHZ], another signal timing that requires attention during the external buffer control is buffer turn-around. The external buffer takes time to reverse the direction of the shared I/O bus. This situation is more relevant when a read is followed by a write. To handle this, GCR[TBCTL\_TRN\_TIME] is defined, which represents the time for which BCTL remains high before starting another flash access. The timings are shown in the following figures.



**Figure 25-18. BCTL signal in NAND read followed by any other operation**



**Figure 25-19. BCTL signal in NOR read followed by any other operation**

## 25.4.6 External transceiver enable (TE)

This signal is used to enable/disable the external transceiver depending on whether a slow/fast memory is connected to the bank servicing the current request.

If various fast/slow memories are connected to different banks of the IFC, the AD bus may become loaded due to sharing. Such a scenario results in poor rise/fall times and therefore limits the speed of operation of fast memories given their strict timing requirements. To address this issue, fast memories should be segregated from the slower memories using an external transceiver that is connected to the AD bus. In these scenarios, the IFC provides the configurable transceiver enable (TE) pin.

### 25.4.6.1 Transceiver enable during boot

The IFC can be used for booting after reset.

The IFC can obtain the TE value, which is used during booting, in the following ways:

- By sampling the TE pin.

In this scenario, the TE pin acts as an input during reset. During this time, the external transceiver should not drive anything onto the AD bus (that is, keep it tristated). This is accomplished by connecting a weak pull up/down resistor on the TE pin so that the external transceiver is disabled. With `rcw_load/boot_load`, the IFC

samples the TE pin in order to know the polarity of external transceiver's enable pin. With rcw\_load/boot\_load, the value stored in CSPR0[TE] as the default value will be opposite of the polarity of external transceiver's enable pin. Therefore, the polarity of external transceiver's enable pin is configurable with the help of pull up/down resistors.

**Table 25-6. Programming of CSPRn[TE]**

Transceiver Enable (TE) Input	Value of CSPRn[TE]	
	Slow Memory	Fast Memory
Active low	0	1
Active high	1	0

- By the chip booting from fast/slow device

If boot source is a fast device and connected directly to chip (without transceiver), then the chip should drive the IFC\_TE signal such that IFC disables external transceiver.

For booting from slower devices where transceiver is present in the connectivity, then the chip should drive the IFC\_TE signal such that IFC enables external transceivers.

#### 25.4.6.2 Transceiver enable post-boot

After booting, and during normal read/write transactions, the TE pin acts as an output pin.

The configuration of this pin is done on a per-bank basis using CSPRn[TE] bits:

- If a transaction hits bank n, the logic specified by CSPRn[TE] appears on the TE pin when CSn gets selected by a flash interface arbitration.
- If no CSn is selected, the default (board tied) value appears on the TE pin. (By default, the transceiver is disabled.)

This table shows how CSPRn[TE] should be programmed depending on polarity of the transceiver's enable pin and type of memory connected to that particular bank.

**Table 25-7. Programming of CSPRn[TE]**

External transceiver input	Value of CSPRn[TE]	
	Slow memory	Fast memory
Active low	0	1
Active high	1	0

### 25.4.6.3 Transceiver enable example

In this scenario, three chip-selects are used.

The connections are as follows:

- CS0: Slow device
- CS1: Fast device
- CS2: Slow device

Because the transceiver used in this scenario has an active-low enable (TOE), the programming of the CSPRn[TE] should be:

- 0 when a slow device is connected to the corresponding bank (to enable the transceiver).
- 1 when a fast device is connected to the corresponding bank (to disable the transceiver).

Therefore, configuration would be as follows:

- CSPR0[TE] = 0
- CSPR1[TE] = 1
- CSPR2[TE] = 0

## 25.5 NAND flash control machine

The NAND flash control machine (FCM) provides an interface to parallel-bus NAND flash devices.

There are separate machines for asynchronous, NVDDR mode.

### 25.5.1 NAND flash synchronous mode

Only 8-bit port size NAND flash EEPROMs are supported for source Synchronous mode in ONFI 2.2. The commands, address bytes, and data are transferred on AD[0:7].

For both writes and reads, the DQS is used to qualify data. IFC signals CLE and AVD/ALE determine whether the writes are of command type (only CLE asserted), address (only AVD/ALE asserted), or write data (both AVD/ALE and CLE asserted with read/write being high). Reads are indicated by asserting both AVD/ALE and CLE, with read/write being low.

## 25.5.2 NAND flash asynchronous mode

For connections between an 8-bit port size NAND flash EEPROM and the IFC in FCM mode, commands, address bytes, and data are all transferred on AD[0:7] with WE\_B asserted for transfers written to the chip, or RE\_B asserted for transfers read from the chip. IFC signals CLE and AVD/ALE to determine whether writes are of the type command (only CLE asserted), address (only AVD/ALE asserted), or write data (neither CLE nor AVD/ALE asserted).

For connection to a 16-bit NAND flash EEPROM to IFC in FCM mode, the high byte of data appear on AD[0:7], whereas the commands, address bytes, and the low byte of 16-bit data is placed on AD[8:15] during read and write data transfers.

## 25.5.3 NV-DDR Mode

Only 8-bit port size NAND flash EEPROM is supported for NV-DDR.

The commands, address bytes and data are transferred on AD[0:7]. For both writes and reads, the DQS is used to qualify data. IFC signals CLE and ALE determine whether the writes are of command type (only CLE asserted), address (only ALE asserted), or write data (both ALE and CLE asserted with W/R\_B being high). Reads are indicated by asserting both ALE and CLE, with W/R\_B being low.

## 25.5.4 Write protect

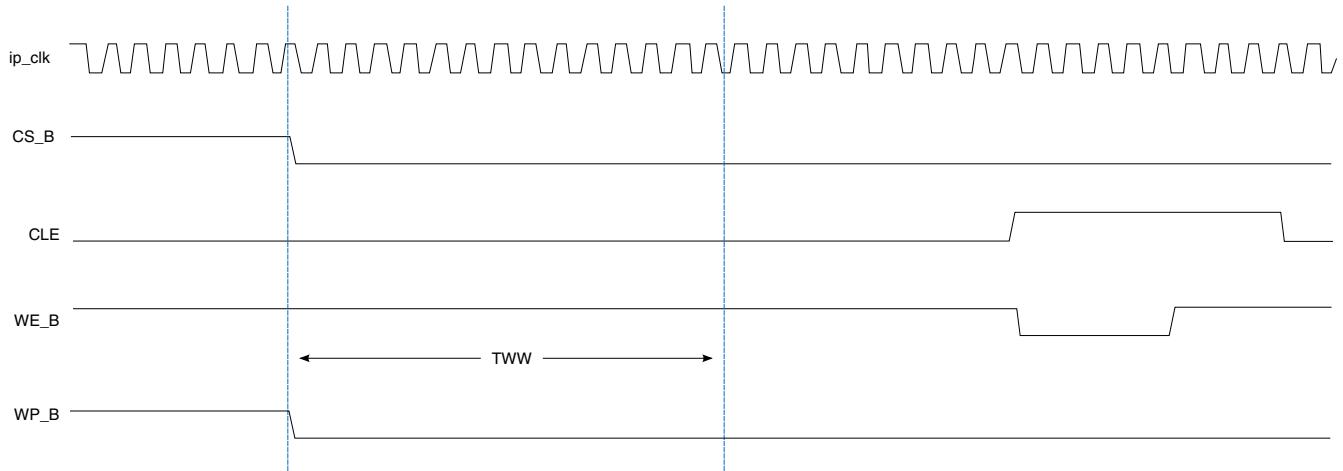
NAND/NOR flash devices come with an input pin, write-protect (WP\_B), used to protect the flash data from any write.

Register bit CSPRn[WP] per chip-select indicates whether or not write accesses are allowed on the particular chip-select.

If a NOR device is connected and a write transaction arrives to a write-protected chip-select, then do not assert chip-select on the NOR flash device and consume the transaction internally. Hence, the protection is controlled by software and the IFC hardware. A write protect error is also generated.

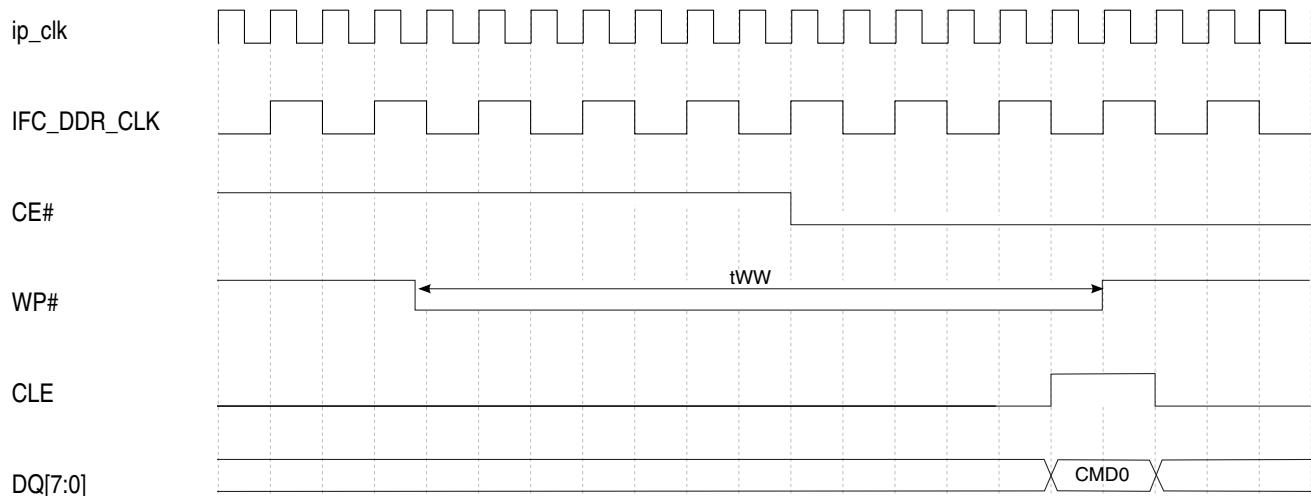
When NAND operation is triggered and IFC detects transition of IFC\_CSPRn[WP] from its previous value then it inserts idle cycles on the interface for TWW time where CS\_B and WP\_B are asserted as shown in figure below.

As a NAND device is not directly memory-mapped like a NOR device and is accessed through the instruction register (FIR), hence there is no prior information of the type of operation to be performed on the NAND. Thus, assert chip-select even if the device is write-protected and expect the device to ignore the write data sent.



**Figure 25-20. Write-protect timing in NAND asynchronous mode**

For synchronous mode, when cleared to zero, the **WP\_B** signal disables the flash array program and erase operations. This signal should only be transitioned while there are no commands executing on the device. After modifying the value of WP, the host should not issue a new command to the device for at least **tWW** delay time. The transition of the **WP\_B** signal is asynchronous and unrelated to any CLK transition in the source synchronous data interface. The following figure shows the write protect timing requirement for source synchronous mode. The figure above shows the timing details.



**Figure 25-21. Write protect timing in NAND (synchronous mode)**

## 25.5.5 SRAM buffer

Read and write accesses to IFC banks controlled by NAND FCM do not access attached NAND flash EEPROMs directly. Rather, these accesses read and write the buffer RAM (a single, shared 16-KB space internal to the IFC and mapped by the base address of every NAND FCM bank).

Even though each NAND FCM-controlled bank has a different base address to differentiate it, all accesses to such banks access the same buffer space. External IFC signals, such as AVD/ALE and CS<sub>n</sub>, do not assert upon accesses to the buffer RAM.

- To perform a page-read operation from a NAND flash device, software initializes the FCM command, mode, and address registers, and triggers the read operation on a particular bank by setting the bit in NANDSEQ\_STRT register. FCM executes the sequence of op-codes held in FIR, reading data from the flash device into the shared buffer RAM. While this read is taking place, software should not access buffer RAM. After loading the complete page on buffer RAM ECC decoding can be performed if it is enabled. Once all the errors have been fixed and if operation completion interrupt is enabled, an interrupt is generated. When FCM has completed its last command, buffer RAM can be read.
- To perform a page-write operation, the software writes data into the SRAM buffer. Then, the software initializes the FCM command, mode, and address registers and triggers the write operation on a particular BANK by setting the bit in NANDSEQ\_STRT register. FCM will execute the sequence of op-codes held in FIR, writing the data from shared buffer RAM to the flash device. While this write is taking place, the software is free to write data in other buffers of the FCM buffer RAM.

By programming NCFGR[NUM\_LOOP] field, the NAND FCM can be used for multi-page read and write operation. A maximum of 16 pages (for 512-byte page), 4 pages (2-KB page), and 2-page (4-KB page) can be used for multi-page operation. Operation completion interrupt will be sent only after completing last page operation.

### 25.5.5.1 Guidelines for SRAM buffer for NAND access

Guidelines for SRAM buffer for NAND access are as follows:

- The IFC supports only one operation at a time on the NAND flash; that is, the user can either read data from the SRAM buffer for a NAND flash read or write data in the SRAM buffer for a NAND flash-write.

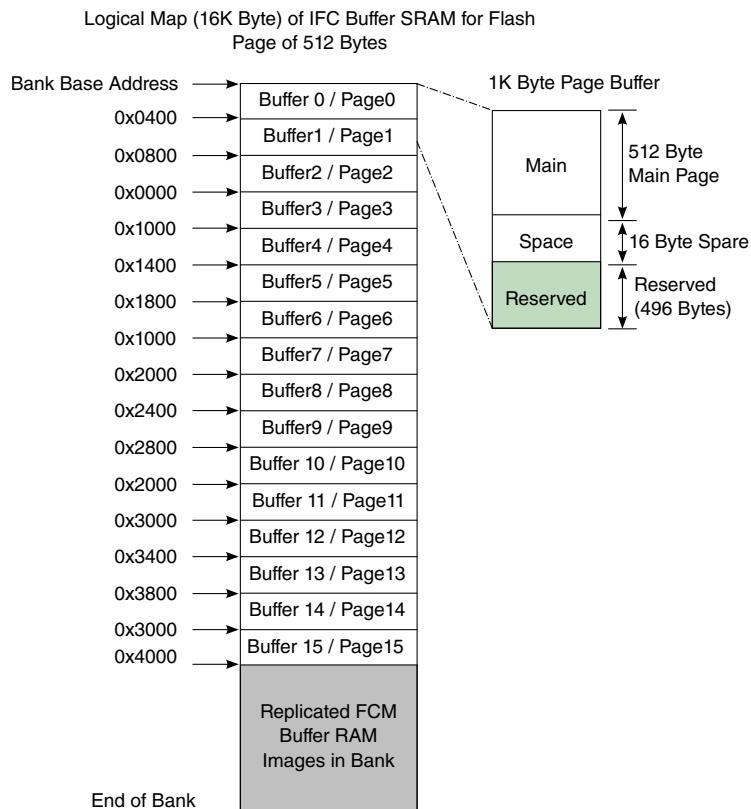
- NAND program: The user must fill the complete program data in the IFC's SRAM buffer before setting the trigger on the IFC to start the data transfer from the SRAM buffer to the NAND flash. When the trigger is set to transfer the data to a NAND device, the SRAM buffer must not be accessed for either read or write. Accessing the SRAM buffer during this time may result in undesirable outcome. The user must wait for the current program operation to complete before accessing the SRAM buffer for the next operation.
- NAND read: When the trigger is set in the IFC, the read data coming from the NAND device is stored in an SRAM buffer. In this operation, the SRAM buffer should not be accessed until the event-completion flag is set.

### 25.5.5.2 Buffer layout and page mapping for 512-byte page NAND flash

The FCM buffer space is divided into 1-KB buffers for 512-byte-page devices ( $\text{CSOR}_n[\text{PGS}] = 00$ ), mapped as shown in the figure below.

The EEPROM's page numbered P is associated with buffer number  $(P \bmod 16)$ , where  $P = \text{ROW}_n$ . Because the bank size set by  $\text{AMASK}_n[\text{AM}]$  is greater than 16 KB, an identical image of the FCM buffer RAM appears replicated every 16 KB throughout the bank address space.

In the case where  $\text{NAND\_BC}[\text{BC}] = 0$ , FCM transfers an entire page, comprising the 512-byte main region followed by the 16-byte spare region; the 496-byte reserved region is not accessed and remains undefined for software. However, for commands given a specific byte-count in  $\text{NAND\_BC}[\text{BC}]$ ,  $\text{COL}_n[\text{MS}]$  locates the starting address in either the main region ( $\text{MS} = 0$ ) or the spare region ( $\text{MS} = 1$ ).



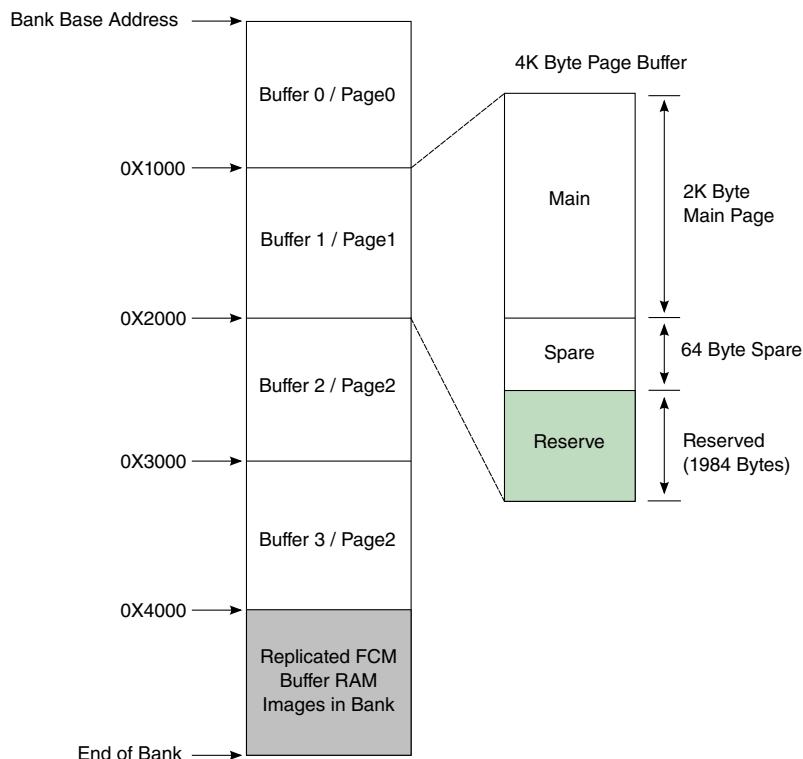
**Figure 25-22. SRAM buffer layout for 512-byte page device**

### 25.5.5.3 Buffer layout and page mapping for 2-KB page NAND flash

The FCM buffer space is divided into four 4 KB buffers for 2-KB page devices (CSOR<sub>n</sub>[PGS] = 01).

Each page in a 2 KB-page NAND flash comprises 2112 bytes, where 2048 bytes appear as main-region data and 64 bytes as spare-region data. The EEPROM's page numbered P is associated with buffer number (P mod 4), where P = ROWN. Because the bank size set by AMASK<sub>n</sub>[AM] is greater than 16 KB, an identical image of the FCM buffer RAM appears replicated every 16 KB throughout the bank address space.

If NAND\_BC[BC] = 0, the FCM transfers an entire page comprising the 2048-byte main region followed by the 64-byte spare region; the 1984-byte reserved region is not accessed, and remains undefined for software. However, for commands given a specific byte count in NAND\_BC[BC], COL<sub>n</sub>[MS] locates the starting address in either the main region (MS = 0) or the spare region (MS = 1). When different IFC banks control both page devices, a 4 KB-page buffer must be assigned to either the first four or last four 512-byte page buffers.



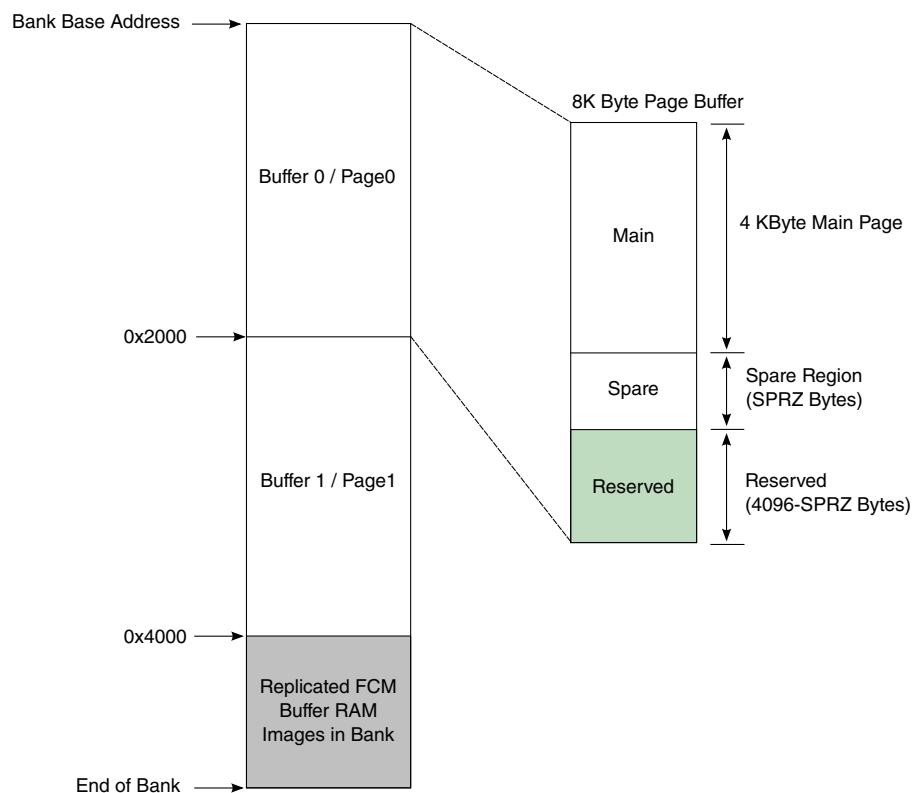
**Figure 25-23. SRAM buffer layout for 2-KB page device**

#### 25.5.5.4 Buffer layout and page mapping for 4 KB page NAND flash

The FCM buffer space is divided into two 8 KB buffers for 4 KB-page devices (CSOR<sub>n</sub>[PGS] = 10).

Each page in a NAND flash comprises 4224 bytes, where 4096 bytes appear as main region data and 128 bytes as spare region data. The EEPROM's page numbered P is associated with buffer number (P mod 2), where P = ROW<sub>n</sub>. Because the bank size set by AMASK<sub>n</sub>[AM] is greater than 16 KB, an identical image of the FCM buffer RAM appears replicated every 16 KB throughout the bank address space.

If NAND\_BC[BC] = 0, FCM transfers an entire page comprising the 4096-byte main region followed by the CSOR<sub>n</sub>[SPRZ]-byte spare region.



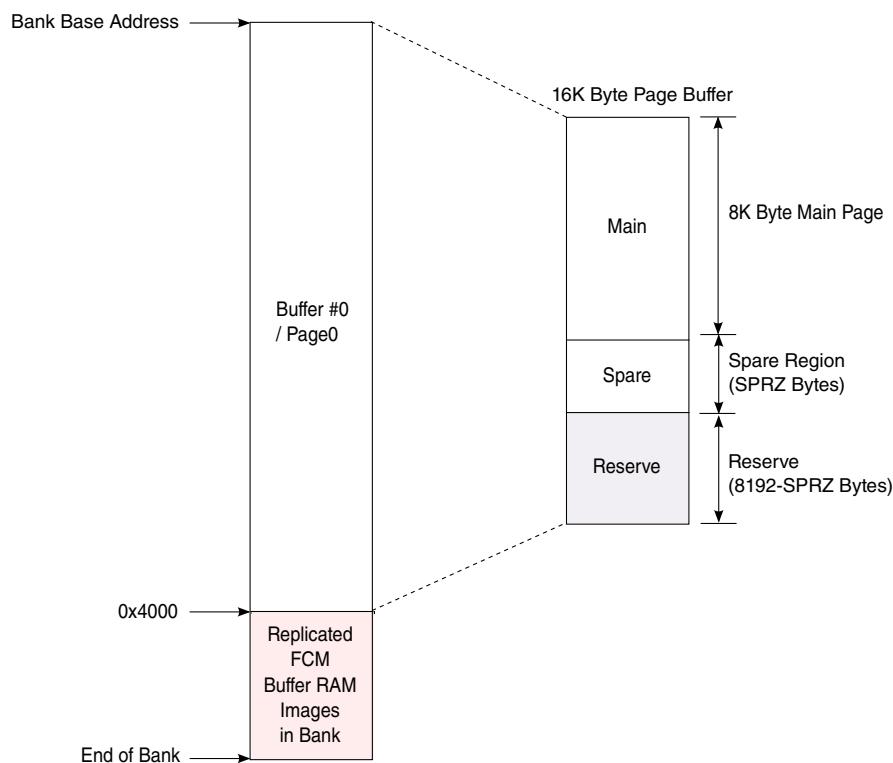
**Figure 25-24. SRAM buffer layout for 4 KB-page device**

### 25.5.5.5 Buffer layout and page mapping for 8 KB page NAND flash

The FCM buffer space is divided into single 8 KB buffers for 8 KB large-page devices (CSORn[PGS] = 11).

Each page in a large-page NAND flash comprises 8192 + spare region bytes, where 8192 bytes appear as main region data, and spare region bytes appear as spare region data. Because the bank size set by AMASKn[AM] will be greater than 16 KB, an identical image of the FCM buffer RAM appears replicated every 16 KB throughout the bank address space.

If NAND\_BC[BC] = 0, the FCM transfers an entire page comprising the 8192-byte main region followed by the CSORn[SPRZ]-byte spare region.



**Figure 25-25. SRAM buffer layout for 8 KB-page device**

### 25.5.5.6 SRAM buffer initialization requirement

Set NCFGR[SRAM\_INIT\_EN] bit to initialize the SRAM before initiating the first program operation. See [NAND Configuration register \(IFC\\_NCFGR\)](#) for details.

### 25.5.6 Use of ECC algorithms

BCH encoder and decoder algorithms are implemented to detect and correct up to 4/8 bits in each 512-byte sector and 24/40 bits in each 1 KB sector.

A Galois Field  $2^{13}$  is used for the encoding and decoding of 4- and 8-bit ECC and a Galois field  $2^{14}$  is used for 24-/40-bit ECC.

#### 25.5.6.1 Generating Galois field elements

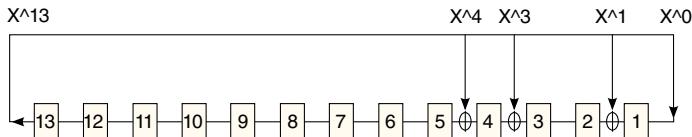
The Galois field elements can be generated by shifting the LFSR.

The following primitive polynomials are used for the computation of respective field elements.

## Galois Field $2^{13}$ field elements

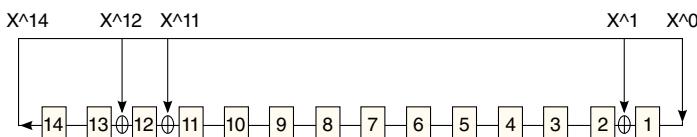
As shown in this figure, the field elements can be generated by shifting the LFSR. Each state of LFSR represents the field element.

$P(x) = 110111000000001$  in binary form and  $1+x+x^3+x^4+x^{13}$  in polynomial form.



## Galois Field $2^{14}$ field elements

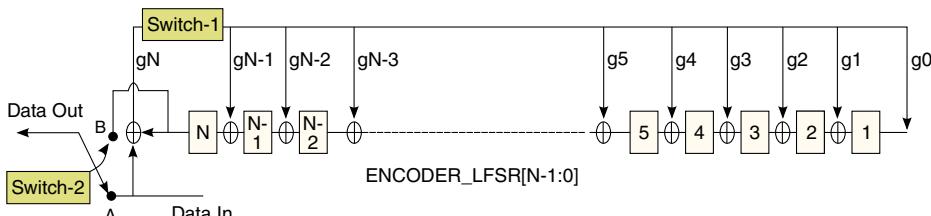
Primitive polynomial  $P(x) = 110000000001101$ ; that is,  $1 + x + x^{11} + x^{12} + x^{14}$



### 25.5.6.2 BCH encoding

This section describes BCH encoding scheme.

This figure represents the encoder implementation.



- 4-bit ECC:  $g(x) = 11010101011000011101010111000010000011000100101000101$
- 8-bit ECC:  

$$g(x)=110001001101111001000111010001110000010111000011100100000110000$$

$$110111100000011100101000100111110101$$
- 24-bit ECC:  

$$g(x)=10100000111010000011010010000010001000100000101100011111011001$$

$$111100111110100001000110000001111000100100010010001000101001010$$

$$110111101010000011100101101110111101101110001100011111011011110$$

$$10011000000111001000000111000010110101111011011001111101110011111011$$

10110011011100010111000110100000000011011000000000110010000110100111  
1

- 40-bit ECC;

$g(x) = 1110000011101100110011110110100101110011101000110000000000$   
 $000111001111000100111000000001010001101110100011101101011101$   
 $01000111001011111000011010110100100000110011000111010001110101$   
 $0011010000011001101101110111010110100011111000010011100100001111$   
 $0100101100101111010011101110110110000101011011100010010011100010$   
 $111111101101001110110100000001010111010100010110011101101010110100$   
 $000010000110011000011001010010111111000110111011001000111011000111$   
 $1111110011100000111101101000101100001111010110011010010100110011$   
 $01111$

In each of the generator polynomial binary forms mentioned above, the msb represents the lowest polynomial power ( $g_0$ ) and the lsb represents highest polynomial power ( $g_N$ ). The following LFSR can be used for BCH encoding.

In encoding, operation data corresponding to a sector is given to the LFSR. During this time, the following events occur:

- Switch 1 is closed and switch 2 is in the down position (A) to allow the transfer of data to the output.
  - After the transfer of a sector's data, switch 1 is opened and switch 2 is moved to the up position (B).
  - Now N parity bits can be read out from the LFSR. ENCODER\_LFSR[N-1] represents the first parity bit out and ENCODER\_LFSR[0] the last parity bit.
    - The value of N is 52, 104, 336 and 560 for 4-, 8-, 24- and 40-bit ECC.

When the IFC accesses the NAND flash device for a read operation, it performs a BCH detection algorithm and indicates how many bit errors were detected and corrected. The number of errors per sector gets registered in the ECCSTAT0/1/2/3 registers. For 24- and 40-bit ECC, only ECCSTAT0/1 are valid as only 8 sectors of 1 KB is possible for SRAM buffer.

ECC is kept in spare region of page at offset 08h. BCH ECC bytes are first written to the SRAM buffer and then to the NAND flash device. During program, the user can read the ECC bytes by reading the appropriate locations in the SRAM buffer.

During encoding, ECC is always calculated for 512-byte data for 4-/8-bit ECC and 1 KB data for 24-/40-bit ECC. CSORn[ECC\_MODE] can be used to select 4-/8-/24-/40-bit correction mode, and CSORn[ECC\_ENC\_EN] and CSORn[ECC\_DEC\_EN] can be used to enable/disable the ECC logic.

## NAND flash control machine

If the encoder is enabled to calculate the ECC, then the data to the encoder is fed in the following manner:

- 16 bits are fed to the encoder every other clock cycle
- The encoder processes the bits in the following manner:
  - Bit #15 (d0), bit #14 (d1), bit #13 (d2), ..., bit #0 (d15), that is, bit #15 is the first bit to enter the encoder
  - Parity bit #0 (p0) is the first parity bit which comes out of the encoder

This table explains the manner in which the data and parity bits are stored in the 16-bit NAND flash memory for 4-bit encoding/decoding.

### NOTE

- Even when storing data in an 8-bit NAND flash memory, the manner in which the data is fed to the encoder remains unchanged.
- The data to the decoder also needs to be fed in the same manner.

**Table 25-8. 16-bit NAND flash memory organization with 4-bit/sector ECC**

	bit #15	bit #14	bit #13	bit #12	bit #11	bit #10	bit #9	bit #8	bit #7	bit #6	bit #5	bit #4	bit #3	bit #2	bit #1	bit #0
Data Word 0	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15
Data Word 1	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31
...																
Data Word 255	d408 0	d408 1	d408 2	d408 3	d408 4	d408 5	d408 6	d408 7	d408 8	d408 9	d409 0	d409 1	d409 2	d409 3	d409 4	d409 5
Spare Word 0	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
Spare Word 1	s16	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29	s30	s31
.																
Spare Word 3	s48	s49	s50	s51	s52	s53	s54	s55	s56	s57	s58	s59	s60	s61	s62	s63
Spare Word 4	p0	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15
Spare Word 5	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25	p26	p27	p28	p29	p30	p31
.																
Spare Word 7	p48	p49	p50	p51	0	0	0	0	0	0	0	0	0	0	0	0

**Table 25-9. 16-bit NAND flash memory organization with 8-bit/sector ECC**

	<b>bit #15</b>	<b>bit #14</b>	<b>bit #13</b>	<b>bit #12</b>	<b>bit #11</b>	<b>bit #10</b>	<b>bit #9</b>	<b>bit #8</b>	<b>bit #7</b>	<b>bit #6</b>	<b>bit #5</b>	<b>bit #4</b>	<b>bit #3</b>	<b>bit #2</b>	<b>bit #1</b>	<b>bit #0</b>
Data Word 0	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15
Data Word 1	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31
...																
Data Word 255	d408 0	d408 1	d408 2	d408 3	d408 4	d408 5	d408 6	d408 7	d408 8	d408 9	d409 0	d409 1	d409 2	d409 3	d409 4	d409 5
Spare Word 0	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
Spare Word 1	s16	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29	s30	s31
.																
Spare Word 3	s48	s49	s50	s51	s52	s53	s54	s55	s56	s57	s58	s59	s60	s61	s62	s63
Spare Word 4	p0	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15
Spare Word 5	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25	p26	p27	p28	p29	p30	p31
...																
Spare Word 10	p96	p97	p98	p99	p100	p101	p102	p103	0	0	0	0	0	0	0	0
Spare Word 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**Table 25-10. 16-bit, large page (2K) NAND flash memory organization with 4-bit/sector ECC**

	<b>bit #15</b>	<b>bit #14</b>	<b>bit #13</b>	<b>bit #12</b>	<b>bit #11</b>	<b>bit #10</b>	<b>bit #9</b>	<b>bit #8</b>	<b>bit #7</b>	<b>bit #6</b>	<b>bit #5</b>	<b>bit #4</b>	<b>bit #3</b>	<b>bit #2</b>	<b>bit #1</b>	<b>bit #0</b>
Data Word 0	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15
Data Word 1	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31
...																

Table continues on the next page...

**Table 25-10. 16-bit, large page (2K) NAND flash memory organization with 4-bit/sector ECC (continued)**

Data Word 1023	d163_68	d163_69	d163_70	d163_71	d163_72	d163_73	d163_74	d163_75	d163_76	d163_77	d163_78	d163_79	d163_80	d163_81	d163_82	d163_83
Spare Word 0	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
Spare Word 1	s16	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29	s30	s31
.																
Spare Word 3	s48	s49	s50	s51	s52	s53	s54	s55	s56	s57	s58	s59	s60	s61	s62	s63
Spare Word 4	p0_0	p0_1	p0_2	p0_3	p0_4	p0_5	p0_6	p0_7	p0_8	p0_9	p0_10	p0_11	p0_12	p0_13	p0_14	p0_15
Spare Word 5	p0_16	p0_17	p0_18	p0_19	p0_20	p0_21	p0_22	p0_23	p0_24	p0_25	p0_26	p0_27	p0_28	p0_29	p0_30	p0_31
.																
Spare Word 7	p0_48	p0_49	p0_50	p0_51	0	0	0	0	0	0	0	0	0	0	0	0
Spare Word 8	p1_0	p1_1	p1_2	p1_3	p1_4	p1_5	p1_6	p1_7	p1_8	p1_9	p1_10	p1_11	p1_12	p1_13	p1_14	p1_15
Spare Word 9	p1_16	p1_17	p1_18	p1_19	p1_20	p1_21	p1_22	p1_23	p1_24	p1_25	p1_26	p1_27	p1_28	p1_29	p1_30	p1_31
.																
Spare Word 11	p1_48	p1_49	p1_50	p1_51	0	0	0	0	0	0	0	0	0	0	0	0
Spare Word 12	p2_0	p2_1	p2_2	p2_3	p2_4	p2_5	p2_6	p2_7	p2_8	p2_9	p2_10	p2_11	p2_12	p2_13	p2_14	p2_15
Spare Word 13	p2_16	p2_17	p2_18	p2_19	p2_20	p2_21	p2_22	p2_23	p2_24	p2_25	p2_26	p2_27	p2_28	p2_29	p2_30	p2_31
.																
Spare Word 15	p2_48	p2_49	p2_50	p2_51	0	0	0	0	0	0	0	0	0	0	0	0
Spare Word 16	p3_0	p3_1	p3_2	p3_3	p3_4	p3_5	p3_6	p3_7	p3_8	p3_9	p3_10	p3_11	p3_12	p3_13	p3_14	p3_15
Spare Word 17	p3_16	p3_17	p3_18	p3_19	p3_20	p3_21	p3_22	p3_23	p3_24	p3_25	p3_26	p3_27	p3_28	p3_29	p3_30	p3_31

Table continues on the next page...

**Table 25-10. 16-bit, large page (2K) NAND flash memory organization with 4-bit/sector ECC (continued)**

Spare Word 19	p3_4 8	p3_4 9	p3_5 0	p3_5 1	0	0	0	0	0	0	0	0	0	0	0	0	0
Spare Word 20																	
...																	
Spare Word 31																	

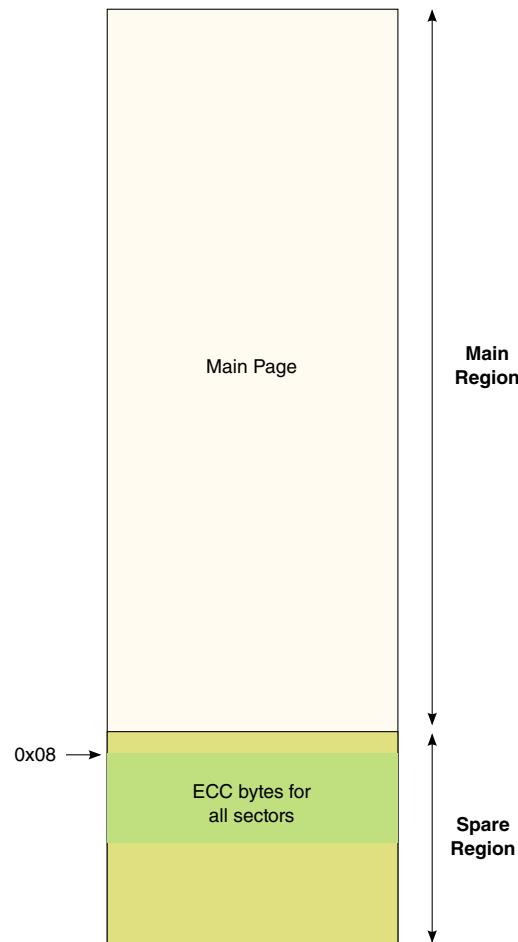
Note that pn\_m, such that

- n = sector number (0, 1, 2, 3), where:
  - Sector 0: d0 to d4095
  - Sector 1: d4096 to d8191
  - Sector 2: d8192 to d12287
  - Sector 3: d12288 to d16383
- m = parity bit number, where:
  - 4-bit ECC: 0 to 51 (12 bits padded to 0 to align the number of parity bytes to the 8 bytes boundary)
  - 8-bit ECC: 0 to 104 (24 bits padded to 0 to align the number of parity bytes to the 8 bytes boundary)

During decoding, the full page is loaded into buffer RAM from flash and then decoding begins. This is required because the BCH decoder can work when a sector's data and corresponding ECC bytes are supplied to the decoder.

For 4-bit correction, 8 parity bytes, 8-bit correction 16 parity bytes, 24-bit correction 42 parity bytes, and 40-bit correction 70 parity bytes per sector are required. These parity bytes are stored in the spare region of the page at offset 08h. For small pages, only 4-bit mode is allowed. For a 2 KB page, four sectors of 512 bytes each can be present in the main region; hence a total of  $4 \times 8 = 32$  parity bytes are stored at offset 08h. For a 4 KB page size, eight sectors of 512 bytes each can be present; hence  $8 \times 8 = 64$  parity bytes are required for 4-bit mode and 128 bytes for 8-bit mode.

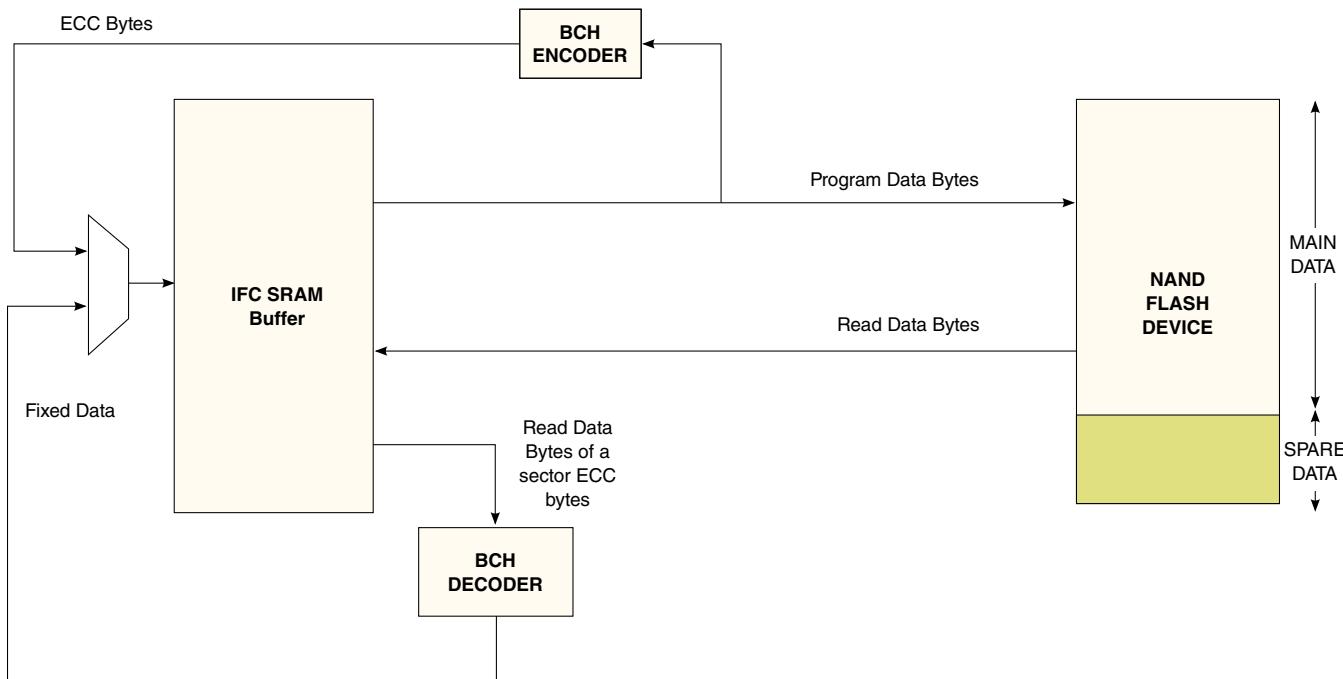
In 24- and 40-bit mode, the parity information is stored adjacent to each other in the spare region; that is, the ECC bytes will be placed one after another. With 24-/40-bit ECC mode, there is no need to pad 0 bits in order to align it to the 8-byte boundary, unlike the padding that is performed with the 4-/8-bit ECC mode.



**Figure 25-26. ECC arrangement on a page**

For example, there is a memory of 4 KB page size and a 24-bit ECC is needed to protect it. A 4 KB page size memory will have four sectors of 1 KB each. The encoder will generate 42 bytes of parity information for each sector. Therefore, there will be a total of  $42 \times 4 = 168$  ECC bytes. All 168 bytes are placed in the spare region of the NAND flash at offset 08h: { Sector-1 {P0, P1,...P335}, Sector-2 {P0,P1,...P335}, Sector-3 {P0, P1,...P335}, Sector-4 {P0,P1,...P335} } from offset 08h where P0 is the first parity bit coming out from encoder and P335 the last bit. The data and parity feeding and arrangement for all the modes remains same and as shown in [Table 25-10](#).

This figure represents the logical flow of encoding and decoding operation during program and read. During program, ECC bytes are also written back in SRAM. During read, first, all data bytes corresponding to a page are written in SRAM buffer, and then the decoding starts on sector basis. At any time, only one operation can be performed on the NAND; that is, it can only be either read or programmed.



**Figure 25-27. BCH encoding/decoding during flash program/read**

#### NOTE

The BCH encoder and decoder should be enabled only when performing full-page operations. If there are partial page operations, the encoder and decoder should be disabled.

#### NOTE

The IFC computes the ECC only for the main region data (as the ECC is not computed for the spare region). The decoder detects and corrects 4/8 bits of error on the main region data and the corresponding ECC bytes stored.

### 25.5.7 Programming the NAND FCM

The NAND FCM performs operations on the NAND flash interface based on the values programmed in the common and NAND register spaces.

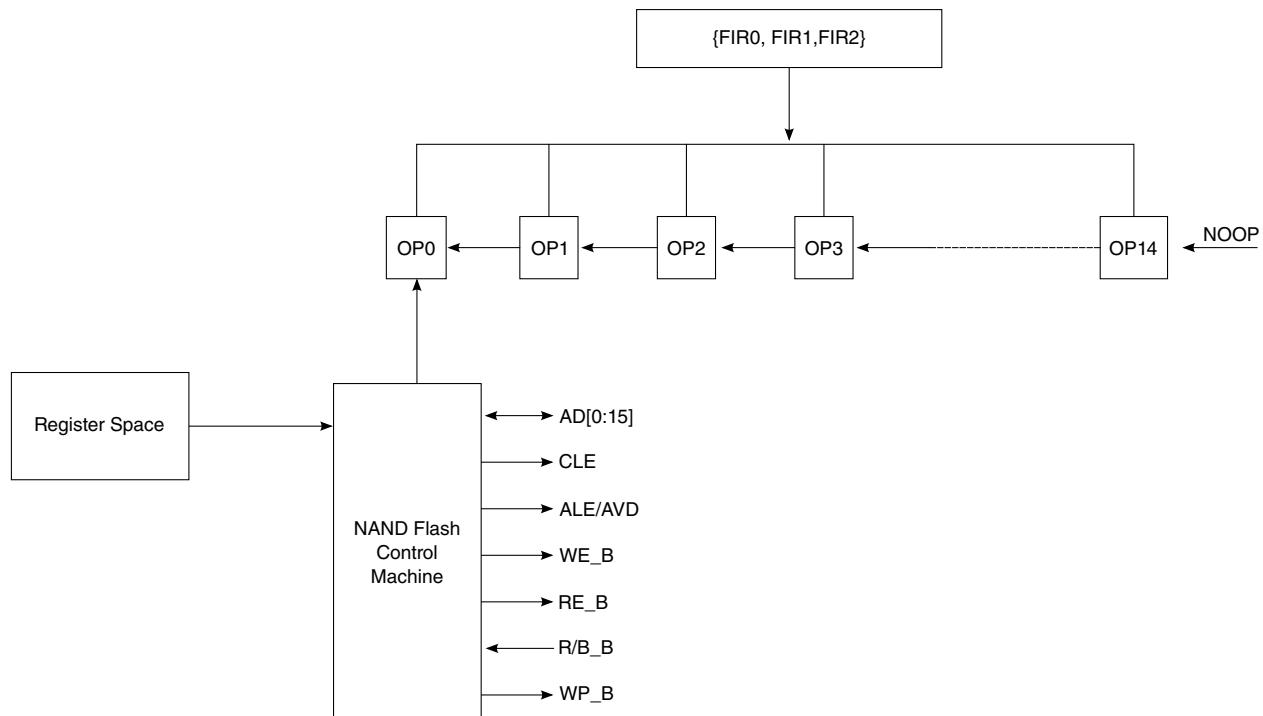
The user is expected to program the following registers with the appropriate values before triggering the NAND FCM operation:

- Chip-select option registers
- Flash timing registers
- General control register
- NAND configuration register
- NAND flash command registers

## NAND flash control machine

- Flash row and column address registers
- NAND flash byte count register
- NAND flash instruction registers
- NAND chip-select register
- NAND event and error interrupt enable register (in case an interrupt needs to be generated)
- NAND control register

When all the above registers have been programmed, the NAND FCM can then be triggered through the NAND operation sequence start register.



**Figure 25-28. FCM instruction sequence mechanism**

### NOTE

User must use the WFR opcode at the end of NAND FIR programming to probe the RDY\_B/BSY\_B signal before end of operation. Status poll is also recommended after every write or erase command.

#### 25.5.7.1 FCM command instructions

There are different types of command instructions.

- Commands that issue immediately (CMD0-CMD7)

- Asynchronous mode: These commands write a single command byte by asserting CLE and WE\_B while driving an 8-bit command on the AD[0:7]/AD[8:15] (8-/16-bit port) bus. The opcode CMD $n$  sources its command byte from the field FCR[CMD $n$ ]. Therefore, up to eight different commands can be issued in any instruction sequence.
- NV-DDR Mode: These commands write a single command byte by asserting CLE with respect to IFC\_DDR\_CLK while driving an 8-bit command on the AD[0:7] bus. The opcode CMD $n$  sources its command byte from the field FCR[CMD $n$ ]. Therefore, up to 8 different commands can be issued in any instruction sequence.
- Commands that wait for RB\_B to be sampled high before issuing (CW0-CW7)
  - Asynchronous mode: These commands first poll the RB\_B signal to be sampled high before writing a single command write on the AD[0:7]/AD[8:15] (8-/16-bit port) bus sourced from FCR[CMD $n$ ] for opcode CW $n$ . It is necessary to use CW $n$  opcodes whenever the memory is expected to be in the busy state (such as following a page read, block erase or program operation) and therefore, initially unresponsive to commands.
  - NV-DDR Mode: These commands first poll the R/B signal to be sampled high before writing a single command write on the AD[0:7] (8 bit port) bus sourced from FCR[CMD $n$ ] for opcode CW $n$ . It is necessary to use CW $n$  opcodes whenever the memory is expected to be in the busy state (such as following a page read, block erase or program operation) and therefore, initially unresponsive to commands.

Consult the manufacturer's datasheet to determine the values to be programmed into the FCR register and whether a given command in the sequence is expected to initiate device busy behavior.

### 25.5.7.2 FCM address instructions

Address instructions are used to issue addresses to the NAND flash device.

- Asynchronous mode: A complete address is formed by a sequence of 1 or more bytes, each written onto AD[0:7]/AD[8:15] (8-bit/16-bit port) with AVD/ALE and WE\_B asserted together.
- NV-DDR Mode: A complete address is formed via a sequence of one or more bytes, each written onto AD[0:7] (8-bit port) with AVD/ALE asserted with respect to IFC\_DDR\_CLK.

### 25.5.7.3 FCM data read instructions

Read data instructions assert RE\_B repeatedly to transfer one or more bytes of read data from the NAND flash device.

The different types of data read instructions provided are as follows:

- Read BC bytes of data from the NAND flash device into the internal SRAM
  - Asynchronous mode: This instruction reads NAND\_BC[BC] into the current FCM SRAM buffer addressed. The data driven by the NAND flash device is sampled on the basis of the TRAD timing parameter and the RE\_B is asserted for a time period equivalent to the TRP timing parameter. If NAND\_BC[BC] = 0, an entire page (including the main and the spare regions) is transferred in a burst.
  - NVDDR mode: This instruction reads NAND\_BC[BC] into the current FCM SRAM Buffer addressed. The data driven by the NAND flash device is sampled on the basis of the quarter cycle shifted DQS . If NAND\_BC[BC] = 0, an entire page (including the main and the spare regions) is transferred in a burst.
- Read BC bytes of data from the NAND flash device into the internal SRAM during boot
  - Asynchronous mode: This instruction reads NAND\_BC[BC] into the current FCM SRAM buffer addressed. The data driven by the NAND flash device is sampled on the basis of the TRAD timing parameter and the RE\_B is deasserted once the read data has been sampled by the NAND FCM. If NAND\_BC[BC] = 0, an entire page (including the main and the spare regions) is transferred in a burst. This instruction can also be used for non-boot applications wherein the RE\_B signal needs to be asserted until the read data has been sampled.
  - NV-DDR Mode: Not valid
- Read the NAND flash device status
  - Asynchronous mode: This instruction performs a status read operation on the NAND flash device. The status returned by the device is stored in the NAND\_FSR register.
  - NVDDR mode: This instruction performs a status read operation on the NAND flash device. The status returned by the device is stored in the NAND\_FSR register. Two bytes (duplicate) of data will be stored in the NAND\_FSR register.
- Read data into the MDR register
  - Asynchronous mode: This instruction reads 8/16 bits of data from a given page in the NAND flash device (based upon the column address programmed) into the MDR register.
  - NVDDR mode: This instruction reads 16 bits of data from a given page in the NAND flash device (based upon the column address programmed) into the MDR register.

#### 25.5.7.4 FCM data write instructions

Write data instructions assert WE repeatedly to transfer program data to the NAND flash device.

- Asynchronous mode: Write data instructions assert WE repeatedly to transfer 1 or more bytes of program data to the NAND flash device. If NAND\_BC[BC] = 0, an entire page (including the main and the spare regions) is transferred in a burst.
- NV-DDR Mode: Write data instructions assert CLE and ALE together to transfer 2 or more bytes of program data to the NAND Flash Device. The data is driven along with centre aligned DQS for the NAND to sample the data. If NAND\_BC[BC] = 0, an entire page (including the main and the spare regions) is transferred in a burst.

#### 25.5.8 Looping FIR sequences

The loop feature provides the flexibility to loop certain FIR sequences (such as program, read, and erase) a finite number of times based on the value programmed in NCFGR[NUM\_LOOP].

The row and column addresses sent to the NAND flash device during a loop operation depend on NCFGR[ADDR\_MODE]. For a more detailed description of the addresses performed on the NAND flash device for every subsequent execution of the FIR sequence during a loop operation, see [NAND Configuration register \(IFC\\_NCFGR\)](#).

##### NOTE

For this feature to correctly execute, it is important to note that the opcodes programmed in the FIR sequence corresponding to the column and row addresses should always be CA0 and RA0. Therefore, it is recommended that only one type of operation (such program/read/erase) be performed through a loop sequence.

#### 25.5.9 NAND DLL

The DLL is a dynamically-adaptive clock delay module. It provides the ability to programmably select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

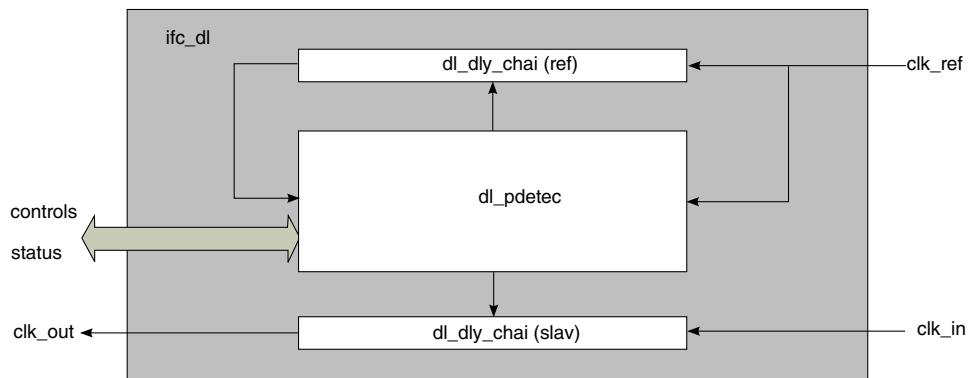
This module is functional when NVDDR device is connected to IFC.

- The DLL is used to calculate the half pulse width of an internally generated reference clock (clk\_ref) and has the same period as IFC\_ND\_DDR\_CLK. This calculated half pulse width is used to calculate a delay for the incoming raw DQS (clk\_in), which is then used to capture the read data

The DLL is comprised of three major components:

- One instance of the delay chain - used for reference calibration (half-phase detect)
- One instance of the delay - used as the slave delay line (performs the programmed delay)
- Phase detector module - forms the control loop with the reference delay line and provides the decoding and controls for the slave delay line

The following figure shows the block diagram of the DLL used in the IFC.



**Figure 25-29. DLL block diagram**

The DLL control loop consists of a counter, reference delay line, and phase detector which operate on the ref\_clock reference clock input. The reference clock (clk\_ref) is fed into the reference delay line. After reset, a single delay tap is selected. A phase detector is used to detect the condition where a half shift has occurred. In addition to this, signals are generated to either increment or decrement the counter which controls the delay line (if the half-phase detect condition is not met). Any changes in the delay of the individual elements of the delay chain (due to PVT) will automatically cause the phase detector logic to determine if a change in the counter value is required. Once the half-phase shift is detected, an internal lock signal is generated.

Refer to [DLL configuration guideline \(valid when IFC is in NVDDR Mode\)](#) for guidelines on DLL usage.

## 25.5.10 NAND asynchronous mode timings

This section describes the basic NAND flash timing waveforms and the programmable timing parameters.

[Figure 25-30](#) explains the NAND program cycle; [Figure 25-31](#) and [Figure 25-32](#) explain the basic read cycle waveforms and corresponding timing parameters for NON-EDO and EDO mode flash. Non-EDO mode corresponds to ONFi2.0 async mode 0/1/2/3 while EDO mode corresponds to ONFi mode 4/5.

### 25.5.10.1 NAND asynchronous mode program data timing

This figure shows the NAND flash program.

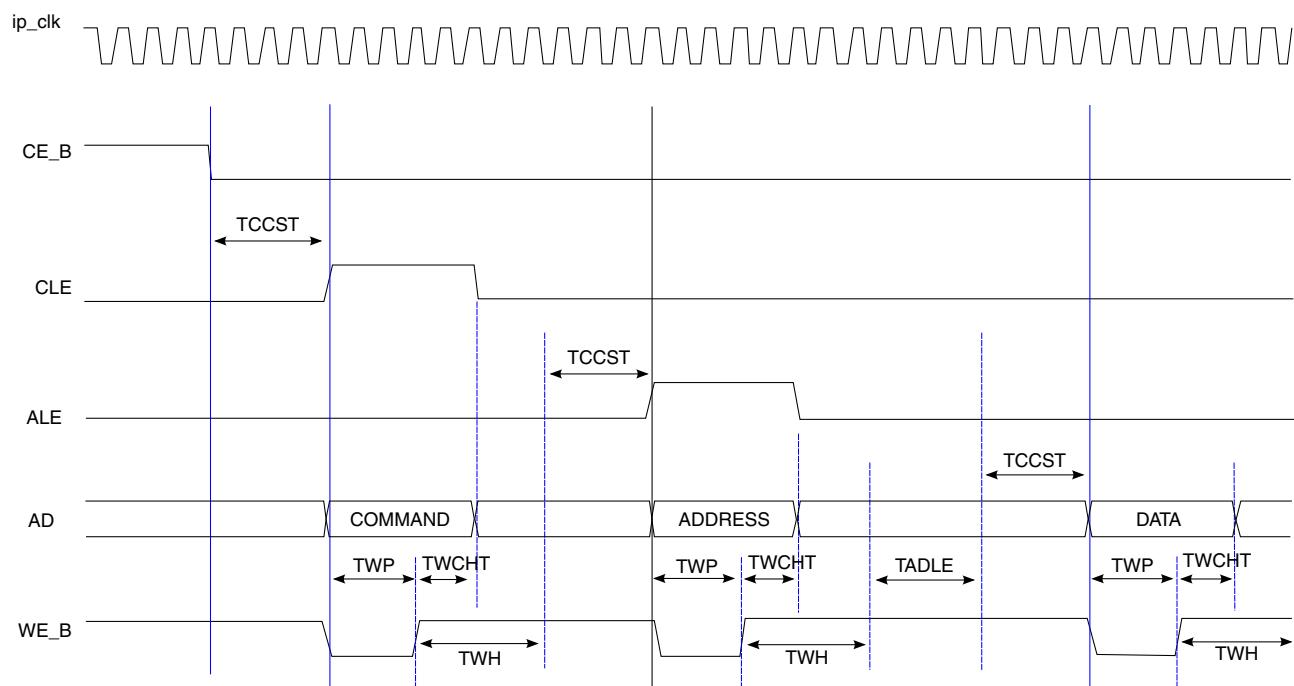


Figure 25-30. NAND flash program

### 25.5.10.2 NAND asynchronous mode read data timing

This figure shows the NAND flash read.

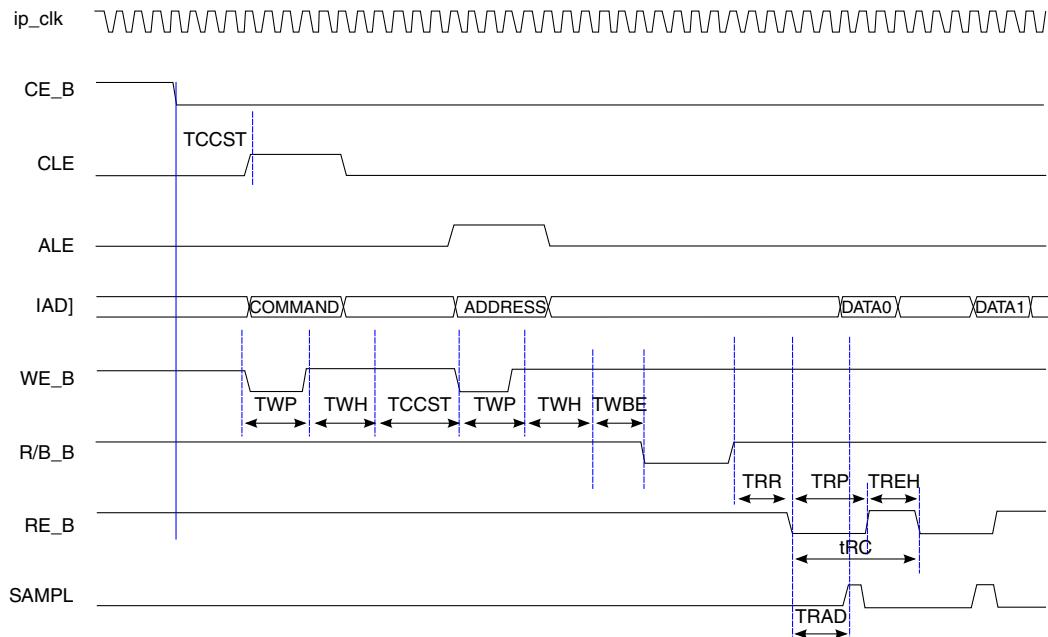


Figure 25-31. NAND flash read

### 25.5.10.3 NAND asynchronous mode calculating read data window width

Calculating the read data window width can be done in several ways.

- Non-EDO mode NAND
  - The window size for the received data is calculated as  $t_{RP} - t_{REA} + t_{RHOH}$ .
- EDO mode NAND
  - The window size for the received data (except the last data beat) is calculated as  $t_{REH} - (t_{REA} - t_{RP}) + t_{RLOH}$ .
  - The window size for last data beat received is calculated as  $t_{RHOH} - (t_{REA} - t_{RP})$ .

Based on these calculations, the minimum size of the data window width is 9 ns (for EDO ONFi mode 5). To support this data window width, a minimum of 333 MHz for the IFC module input clock is required. By increasing the value of timing parameter  $t_{RP}$ , that is, the RE pulse width, the data window size can be increased.

#### NOTE

The read cycle time should satisfy this condition:  $((t_{RP} + t_{REH}) \geq t_{RCmin})$ . If the sum of the  $t_{RPmin}$  and the  $t_{REHmin}$  values specified by the datasheet is less than the  $t_{RCmin}$  (from the datasheet), it is

then recommended that the  $t_{RPmin}$  value be increased to meet the  $t_{RCmin}$  timing requirement. This is recommended because increasing the  $t_{RPmin}$  value causes the data window size to increase.

Read data sampling time (TRAD) represents the read data sampling time on the NAND. The value should be programmed where the sampling at the IFC should be done at the center of the received data eye. Its value can be calculated as [ $T_{REAmx}$  (from NAND datasheet) + 2 x board delay + 1/2 received data window size]. If in EDO mode, the user should choose the TRAD value corresponding to the smaller data window calculated using the two formulae explained above for EDO.

#### **NOTE**

For EDO mode, the value of TRAD should always be less than  $t_{RP} + t_{REH}$ . If the board delay value used in the TRAD calculation results in this condition to not be met, the user may not be able to run the interface at the EDO mode frequency. In this situation, the read timing should be relaxed by increasing  $t_{RP}$  value to access the NAND device at a slower speed.

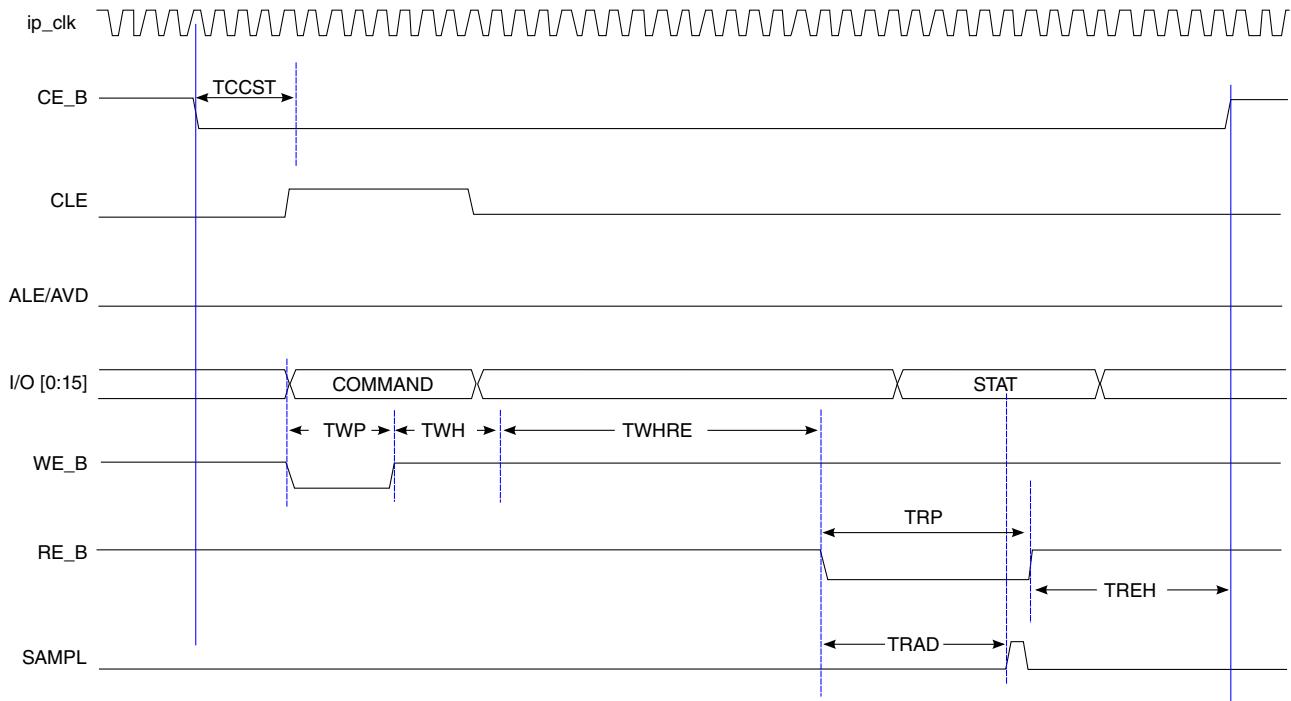
#### **25.5.10.4 NAND asynchronous mode read data sampling approach**

The IFC follows the approach in which the timing parameters are programmable.

The IFC logic runs at the IFC module input clock and can generate these timings on the flash interface through the programmed value in the timing registers. Based on the timing values as given in the ONFi 2.0 spec for each mode, received data window width is calculated (as explained in the previous section).

### 25.5.10.5 NAND asynchronous mode read status timing

This figure shows NAND read status timing.



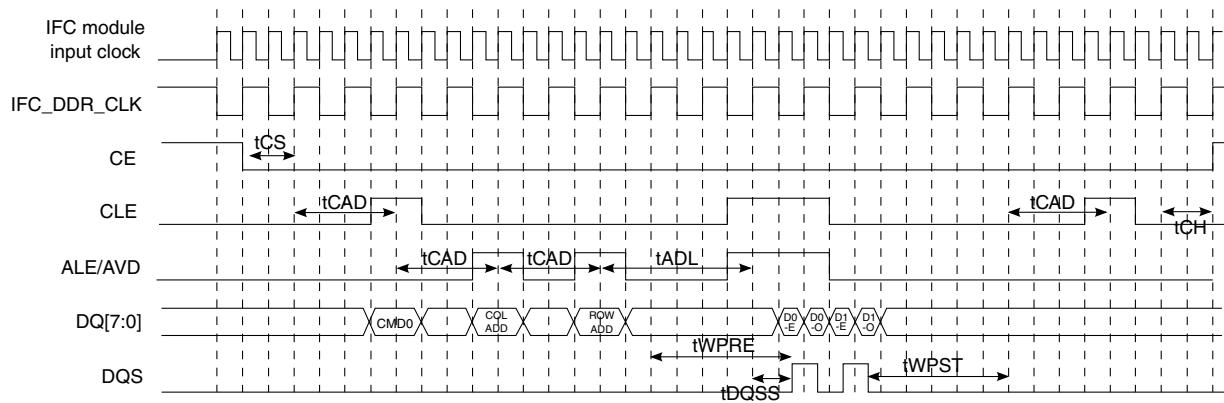
**Figure 25-32. NAND status read**

### 25.5.11 NV-DDR mode timings

#### 25.5.11.1 NAND synchronous mode program operation

In a program operation, the data is expected by the flash memory on both the rising and falling edges of the DQS. The DQS should be center-aligned with respect to the data.

The following figure shows the program operation and the relevant timing parameters for a IFC module input clock:external DDR clock (IFC\_DDR\_CLK) ratio of 1:2.



**Figure 25-33. NAND synchronous mode program operation**

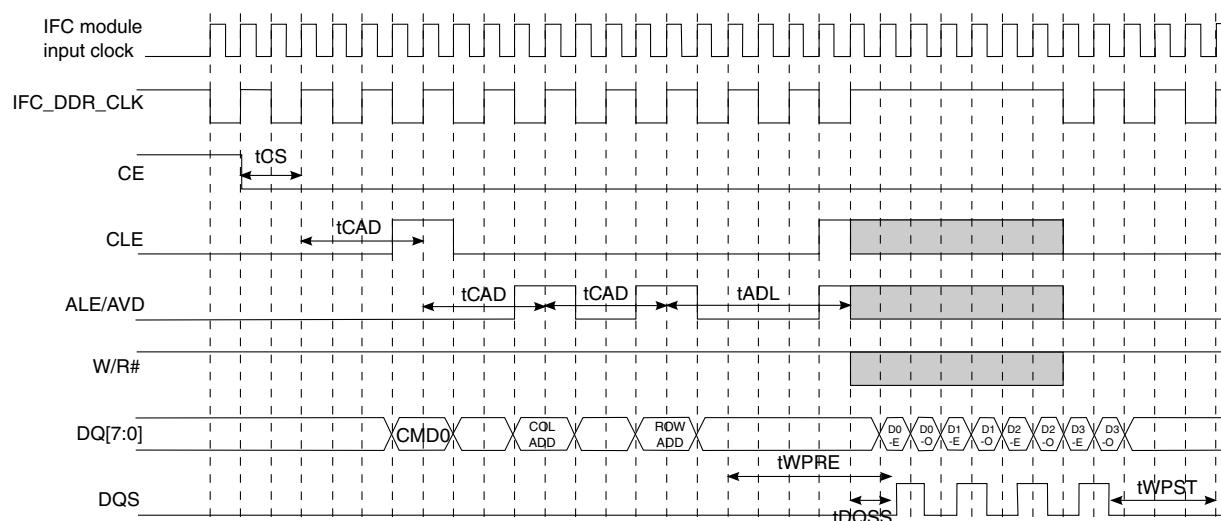
### 25.5.11.2 NAND synchronous mode program with clock stopped

The user can program the IFC to save power during the data input cycles by setting the DDR clock stop bit in the IFC DDR clock control register.

This will result in the IFC\_DDR\_CLK signal being held high (that is, stopping the CLK).

However, the IFC\_DDR\_CLK will be stopped only during program phase. The following figure shows the timing relationship of a program operation with the clock stopped. The values of the AVE/ALE, CLE, and W/R\_n signals are latched on the rising edge of IFC\_DDR\_CLK and thus, while the clock is held high, these signals are don't care.

The following figure shows the program operation with the clock stopped and the relevant timing parameters for an IFC module input clock:external DDR clock (IFC\_DDR\_CLK) ratio of 1:2.



**Figure 25-34. NAND synchronous mode program operation with clock stopped**

### 25.5.11.3 NAND synchronous page read operation

For page read operation, the flash memory sends the read data aligned with the rising and falling edge of the DQS. In this case, the DQS is edge-aligned with the data.

The following figure shows the read page operation and the relevant timing parameters for an IFC module input clock:external DDR clock (IFC\_DDR\_CLK) ratio of 1:2.

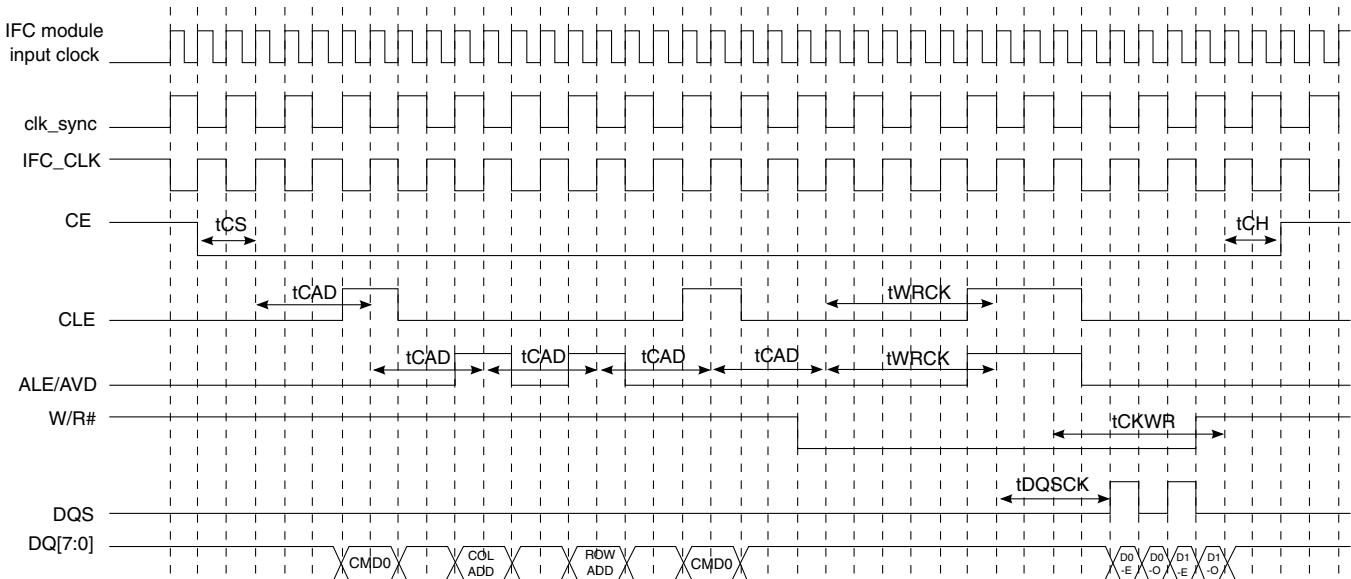


Figure 25-35. NAND synchronous mode page read operation

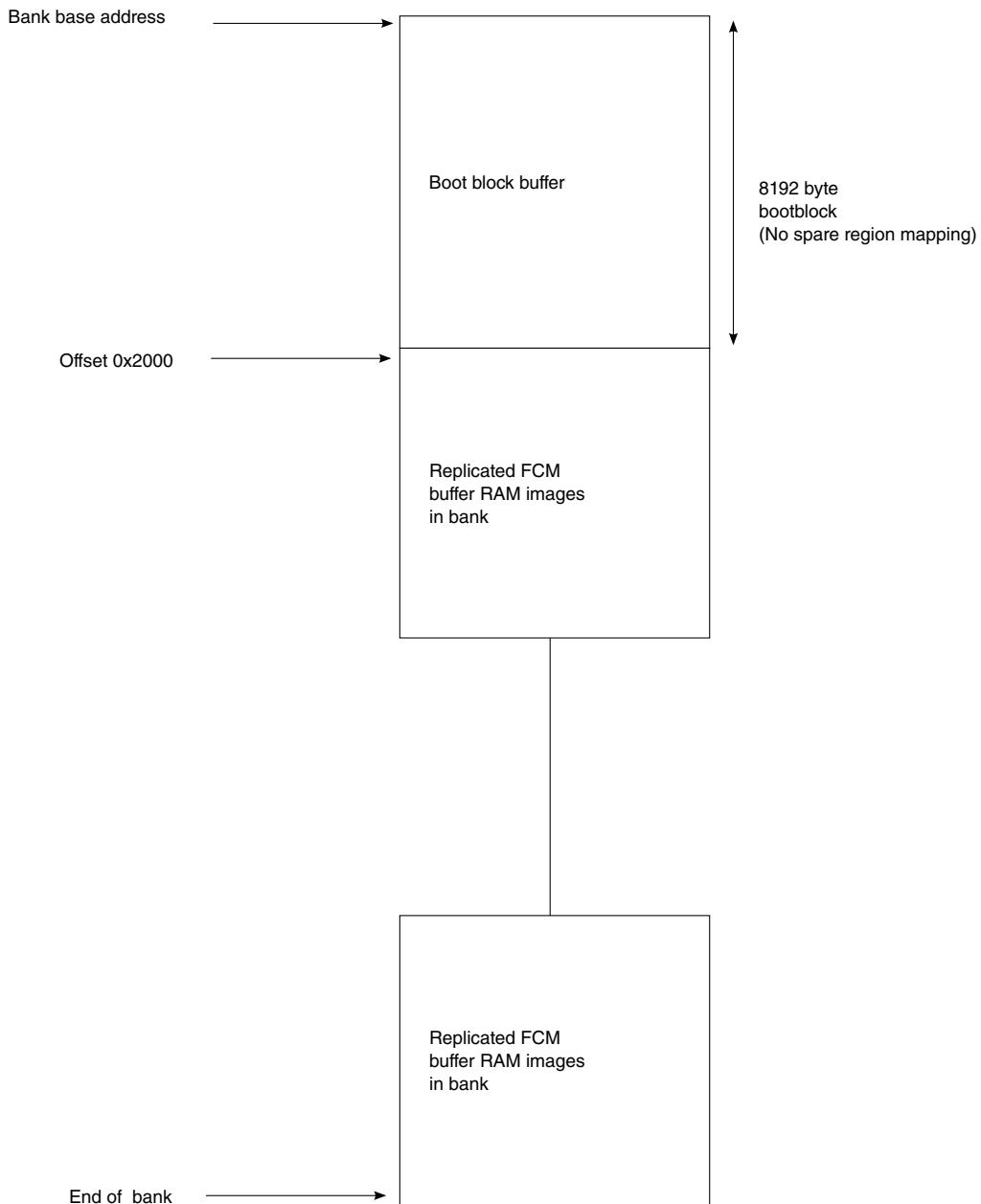
### 25.5.12 NAND asynchronous mode boot mechanism

If the FCM is selected as the boot ROM controller from power-on-reset configuration, the IFC automatically loads up to 8 KB of boot code from BANK0 to the NAND FCM buffer RAM depending on the RCW load or BOOT load indication.

The CPU can execute boot code directly from the FCM buffer RAM, but must ensure that any further data read from the NAND flash EEPROM is transferred under software control in order to continue the bootstrap process.

As AMASK0[AM] is initially cleared during reset, all the CPU fetches to the IFC access the FCM buffer RAM. First, 8 KB of SRAM holds the valid boot data, which appears in the memory map as a 8 KB RAM. No NAND flash spare regions are mapped during boot, therefore only 8 KB of contiguous, main-region data loaded from the first pages of the boot block (good block), are accessible in IFC bank 0, as indicated in this figure. In boot mode, addresses coming to SRAM buffer from system bus wraps at 8 KB physical address boundary.

The software must clear NCFG[BOOT] to enable the logical to physical address mapping.



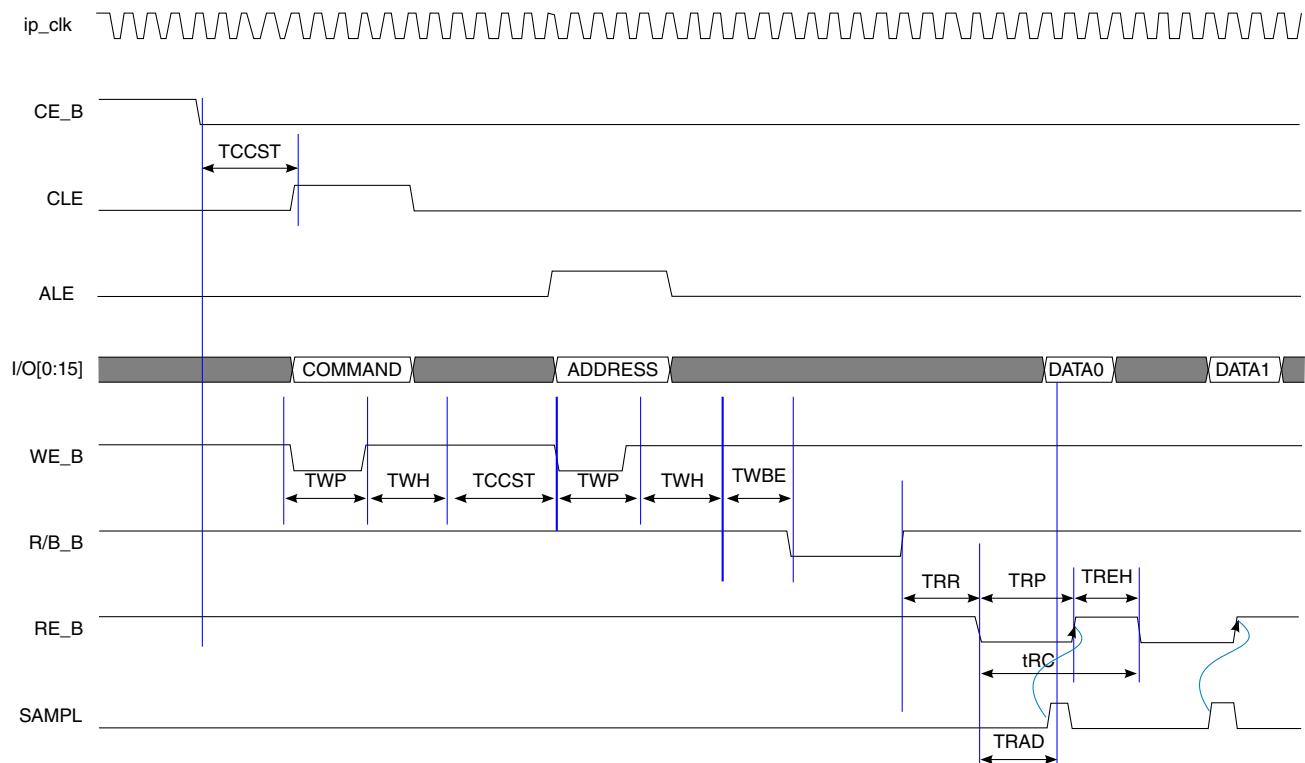
**Figure 25-36. FCM buffer RAM map during boot**

The IFC employs the counter-based read data sampling approach as described earlier. Here, the read data sampling is determined by the counter value whose initial value is determined by the *NAND Flash Specification* and the IFC module input clock , however there can be two different clocks for boot; if the system PLL is not locked, the RCW clock will be coming to the IFC and the counter values for read data sampling will be

governed by this clock frequency; otherwise the IFC would receive the IFC module input clock post-system PLL lock for boot, which results in a different value of the timing counters.

These two clock frequencies are passed as parameters into the IFC during instantiation. Based on this, the IFC initializes timing parameters corresponding to the slowest mode of asynchronous NAND device (ONFi Mode-0) during boot. During normal operation, timing parameters can be programmed as per the device specification. Read data sampling is governed by read access time of NAND flash and board delay. This value is represented by timing parameter FTIM2\_CS0[TRAD]. It can be initialized by the device by passing the TRAD parameter during instantiation.

During boot, the read data sampling mechanism is slightly different than normal read to assure booting. Also, the read-enable signal remains asserted (low) until the data gets sampled, while in normal read operation the read-enable can be de-asserted. This mechanism ensures that there is no case where the read data cannot be sampled by the IFC as data retains its state until the time read enable is asserted.



**Figure 25-37. Read data sampling during boot**

### 25.5.12.1 NAND asynchronous mode boot process

The steps below describe the boot process:

1. After receiving rcw\_load or boot\_load input signals, the IFC commences automatic boot block loading if the FCM is selected as the boot ROM location. Small-page (512-byte page) or large-page (2/4/8-Kbyte page), 8-bit or 16-bit NAND flash devices can be used for boot loading when enabled with CS0. With rcw\_load or boot\_load indication, the CSPR0[WP] bit also is set. With this bit set, the IFC drives WP\_B low on the IFC output during boot accesses to prevent accidental erasure of the NAND flash boot ROM. Software needs to clear the CSPR0[WP] bit to commence normal write operation on CS0 after finishing the booting process.
2. The NAND FCM issues a reset command to the NAND device and then begins searching for a valid boot block from block 0.
3. The NAND FCM reads the spare regions of the first two pages or the first and the last page (depending upon pof\_cfg\_bbi\_srch\_sel) of the current block, checking the bad block indication (BI) bytes to validate the block for reading. BI bytes must hold the value 0xFF for the page to be considered readable.
  - For small-page (512-byte page) devices, the BI is a single-byte read from spare region byte offset 5 (8-bit port size). For a 16-bit port size device, the BI is a single byte read from spare region byte offset 11. The IFC supports booting from a 16-bit port size device only when no ECC is stored in the spare region, that is, the boot block is guaranteed to be good.
  - For large-page (2/4/8-Kbyte page) devices, the BI is a single-byte read from spare region byte offset 0 (8-bit port size), or two-bytes read from spare region byte offsets 0 and 1 (16-bit port size).

If either of the above-mentioned pages of the current block are marked invalid, then the boot block index is incremented by 1 and the FCM repeats step 3. The IFC will continue searching for a bootable block indefinitely; therefore, at least one block must be marked valid for boot loading to proceed. At the conclusion of the boot block search, the value of ROW0 points to the boot block.

4. The FCM optionally checks the ECC at boot time depending on configuration selected during reset (por\_cfg).
5. The FCM reads entire pages from the boot block until /8 Kbytes have been saved to the FCM buffer RAM when booting via the IFC NAND FCM. ECC errors are corrected if possible. If the FCM is unable to correct the ECC errors, the IFC signals an unrecoverable error by asserting the boot\_err signal.
6. The CPU now commences fetching instructions in random order, from the FCM buffer RAM. Boot software must clear NCFGRI1[BOOT] to enable normal operation of FCM.

**NOTE**

During boot, the IFC performs a total of 5/6 address cycles, based on whether the NAND flash device used for booting is a small-page (512-byte page) or a large-page (2/4/8-Kbyte page) device. It is expected that the NAND flash device used to boot will ignore any excess address cycles which it may not need.

The following table describes the hard-coded FIR sequences used during auto-boot operation from the NAND flash.

**Table 25-11. FIR sequences used during boot**

FIR sequence	Description
{CMD3, NOOP}, where CMD3 = 0xFF	Issue soft reset to NAND device
{CW1, CA0, RA0, BTRD, NOOP}, where CW1 = 0x50	Partial page read from small-page device for bad block indicator read (good block search)
{CW0, CA0, RA0}, BTRD, NOOP}, where CW0 = 0x00	Full page read from small-page device from boot page read
{CW0, CA0, RA0, CMD2, BTRD, NOOP}, where CW0 = 0x00, CMD2=0x30	Partial page read from large-page device for bad block indicator read (good block search)
{CW0, CA0, RA0, CMD2, BTRD, NOOP}, where CW0 = 0x00, CMD2= 0x30	Full page read from large-page device from boot page read

**NOTE**

Since fixed values of commands are used for read during auto-boot operations, only NAND flash devices supporting the commands mentioned in the above table can be used for auto-boot read operations.

## 25.6 NOR flash control machine

The NOR FCM allows an interface to asynchronous, simple, and internal latch-based NOR flash devices.

### 25.6.1 NOR boot

For NOR boot, CS0 is the boot chip-select.

Before fetching the first instruction from the NOR flash, the IFC flash timing registers are loaded with default timing parameters.

## 25.6.2 Unmuxed (parallel) asynchronous NOR read timings

The figures show the read cycle and burst read cycle timing diagrams.

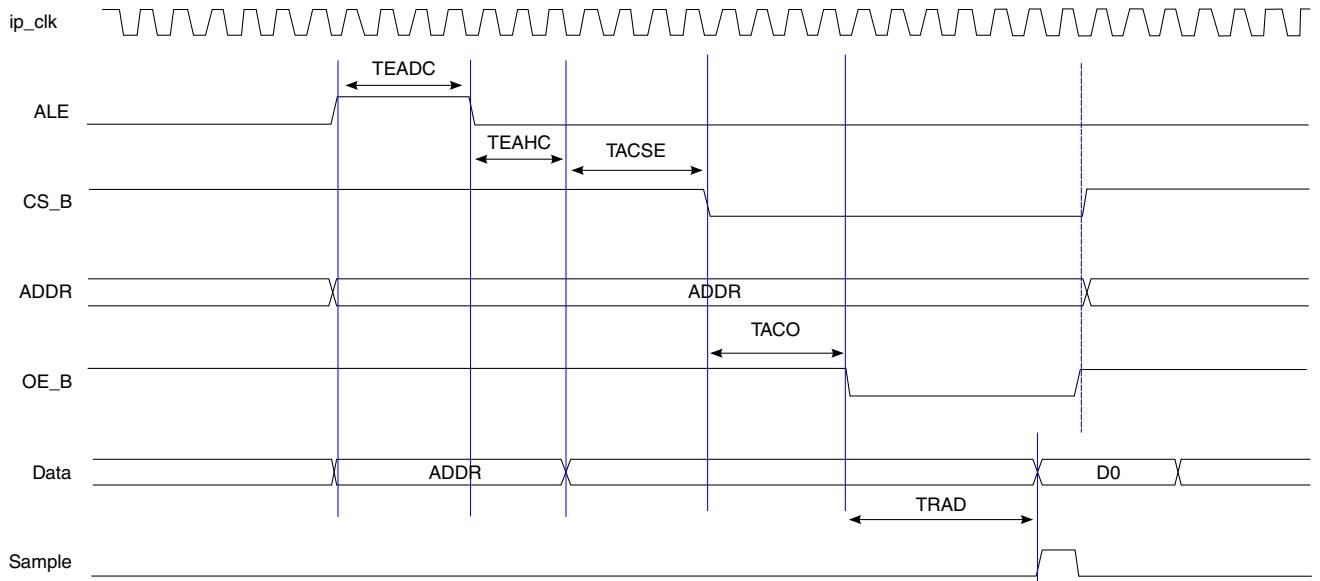


Figure 25-38. Read cycle timing

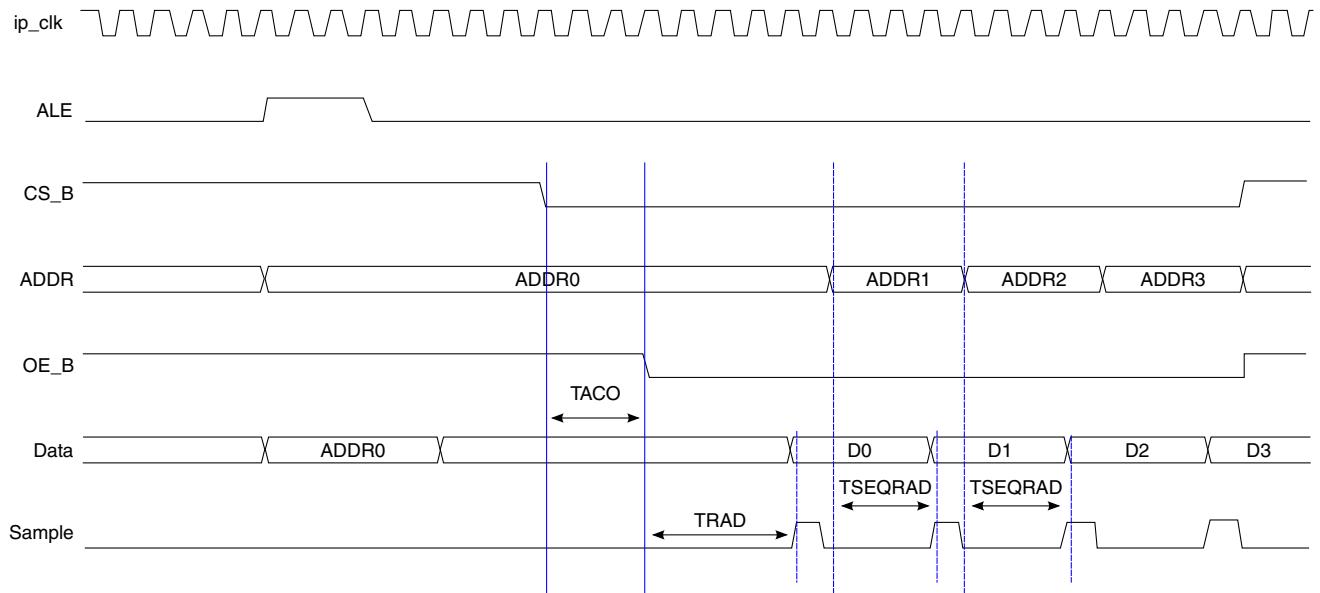


Figure 25-39. Burst read cycle timing

## 25.6.3 Simple asynchronous NOR write timings

The figures show the write cycle and burst write cycle timing diagrams.

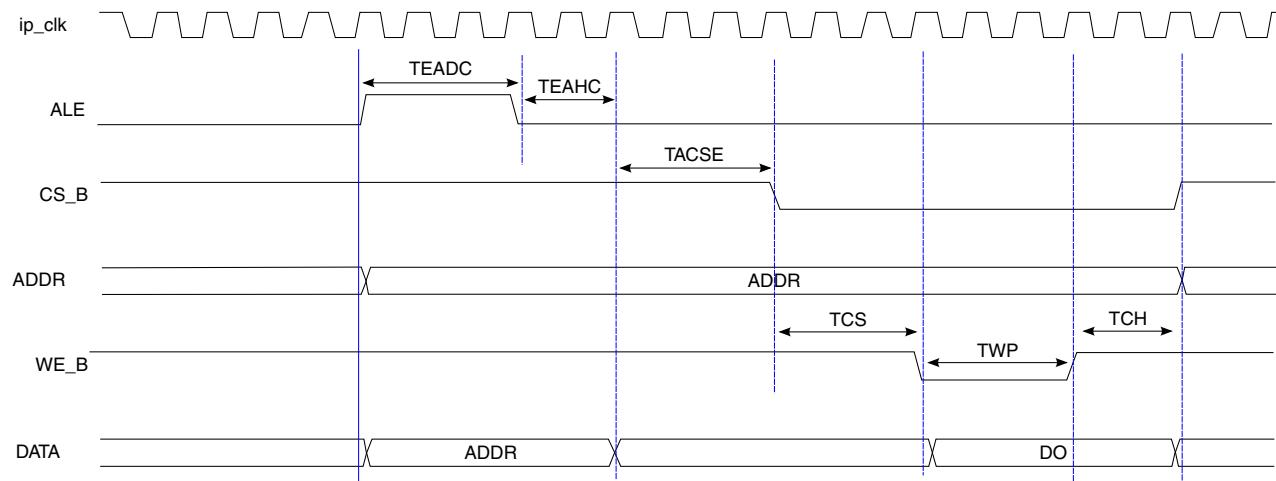


Figure 25-40. Write cycle timing

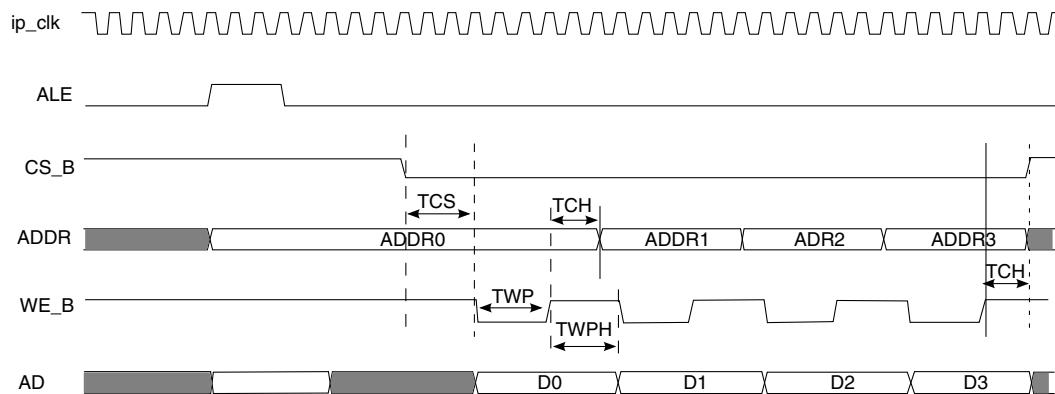
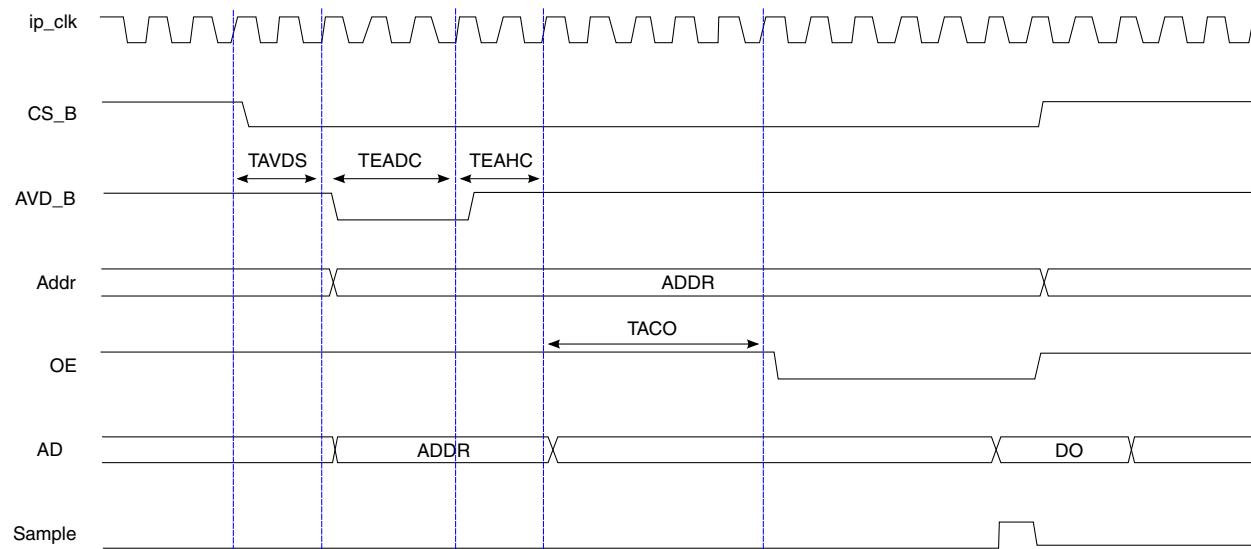


Figure 25-41. Burst write cycle timing

## 25.6.4 Muxed (ADM) asynchronous NOR read timings

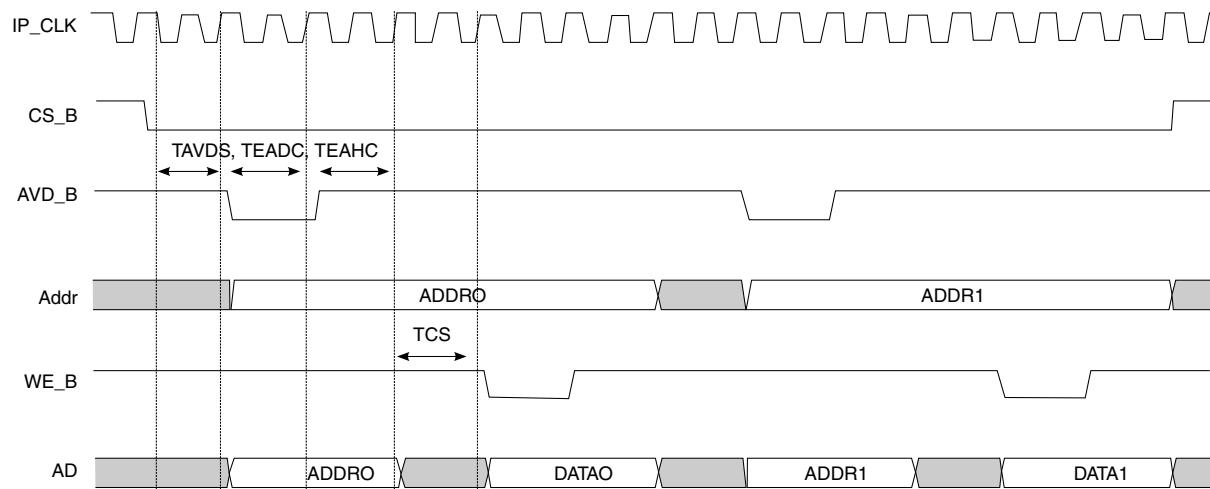
This figure shows the read cycle timing diagram.



**Figure 25-42. Read cycle timing**

## 25.6.5 Muxed (ADM) asynchronous NOR write timings

This figure shows the write cycle timing diagram.



**Figure 25-43. Write cycle timing**

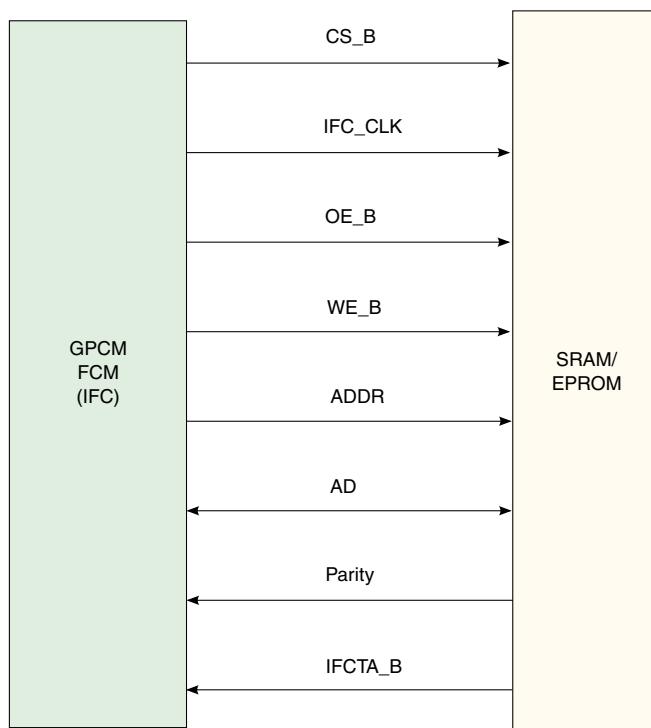
## 25.7 General purpose chip-select machine (GPCM)

The GPCM controller has two modes of operation: normal GPCM and generic ASIC.

Only one mode is operational at a time per chip select. The mode is selected by the value programmed in GPMODE field of CSORn\_GPCM register.

### 25.7.1 Normal GPCM mode of operation

The figure below explains the interface of the IFC to a device connected through the normal GPCM FCM.



**Figure 25-44. Normal GPCM interface**

#### 25.7.1.1 Normal GPCM program operation

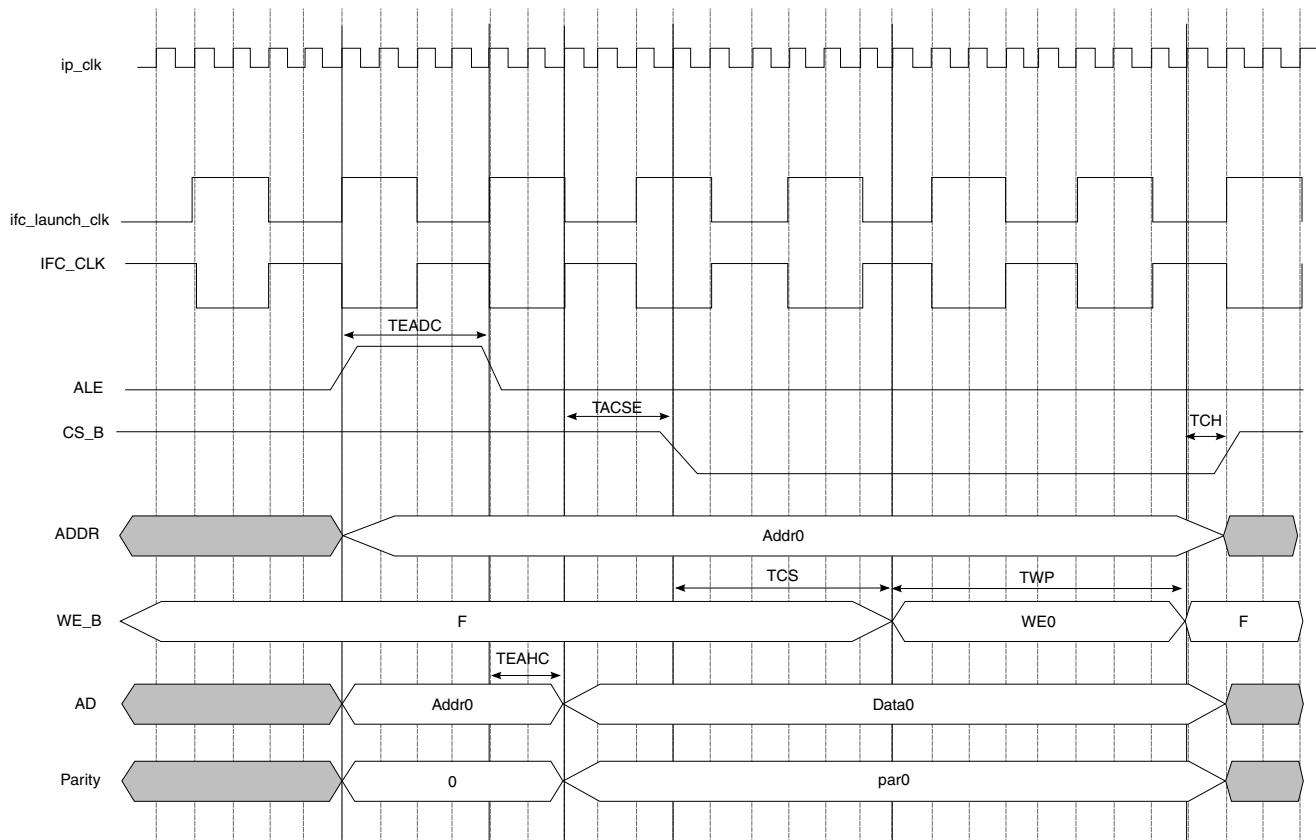
In normal GPCM, the write transaction depends on the register field CSORn[WGETA].

The following options, through an internal counter or through an external device giving an access termination signal, are explained below.

### 25.7.1.1.1 Normal GPCM internal counter-based program operation

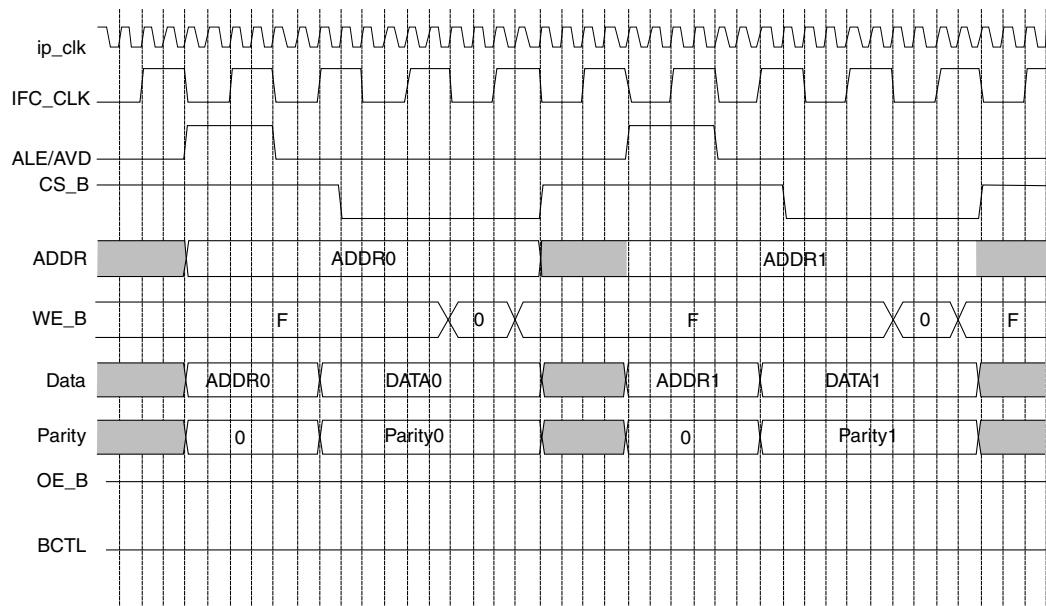
Transactions are based on control-signal timings that are programmed in the flash timing registers for GPCM mode (IFC\_FTIMx\_CS<sub>n</sub>\_GPCM) when CSOrn[WGETA]=0.

Assertion of AVD/ALE is aligned to the rising edge of ifc\_launch\_clk (an internal clock), after that all the timing parameters are in terms of the IFC module input clock (ip\_clk). IFC\_CLK is external clock which goes to the device and it is generated by inverting ifc\_launch\_clk.



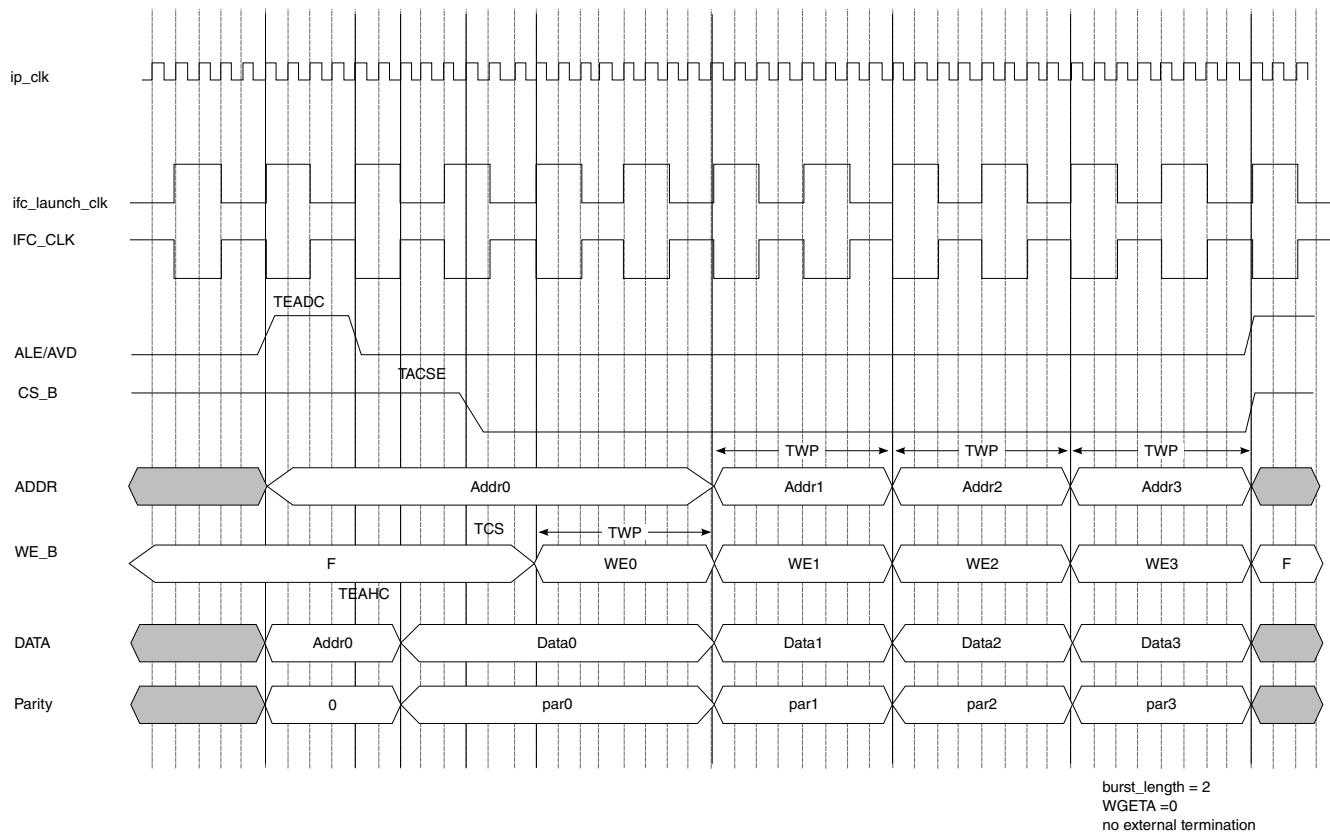
**Figure 25-45. Normal GPCM program operation - non-burst mode**

## General purpose chip-select machine (GPCM)



**Figure 25-46. Normal GPCM back-to-back program operation**

In burst mode, the next data and next address are sent after one TWP period (the amount of time between address/data cycles). If IFCTA\_B is deasserted and the burst is controlled by the timing parameter programmed in FTIM registers, the burst will continue until either the burst length is reached or complete data is transferred. If the number of bytes of data to be transferred is less than programmed burst length multiplied by port size, then the burst will be terminated before the programmed maximum number of burst transfers.



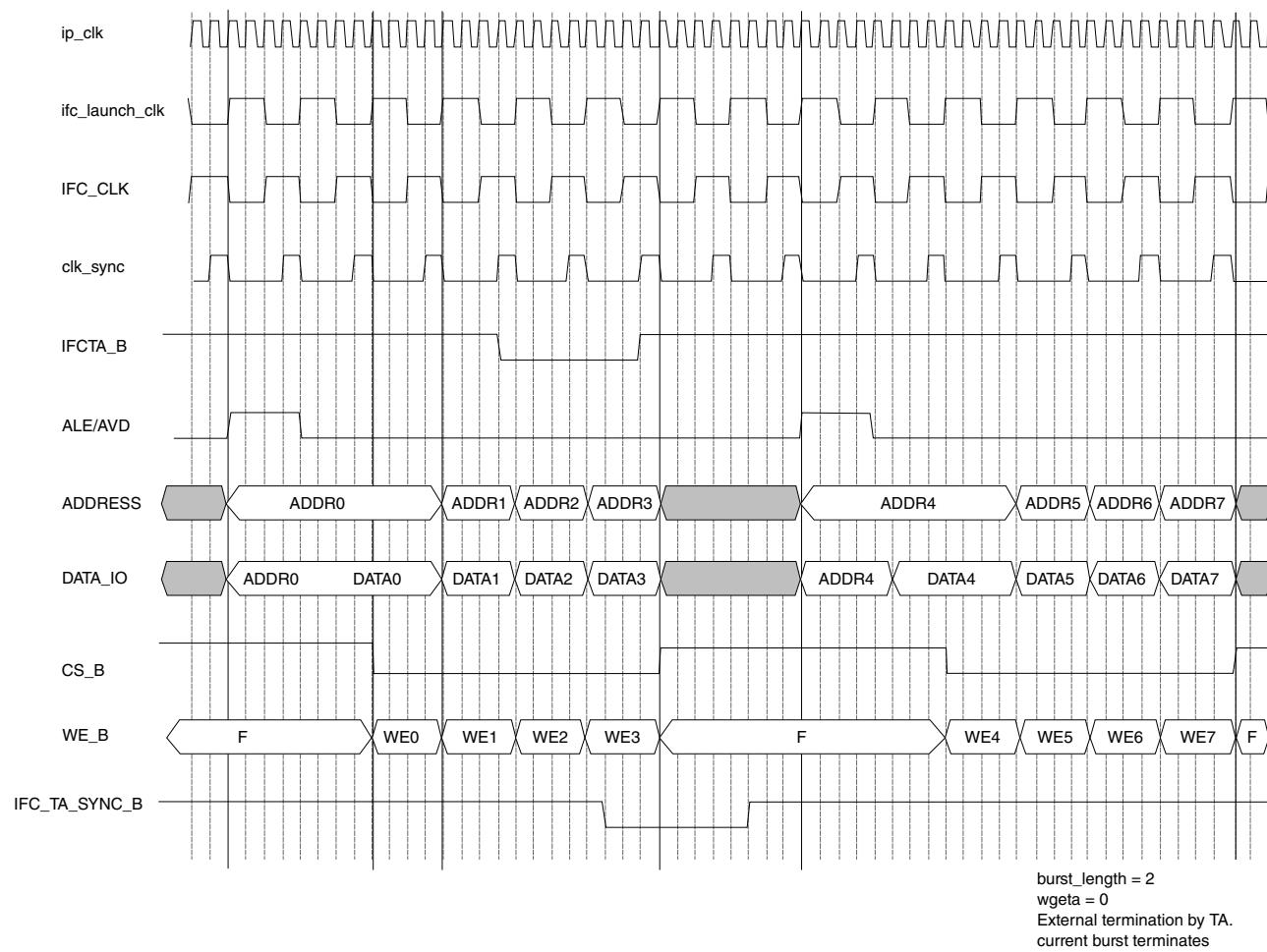
**Figure 25-47. Normal GPCM program operation - burst mode**

For a normal timing-based transaction, **CSORn[RGETA]=0**. If the access is aborted by **IFCTA\_B** and there is remaining data to be transferred, the current burst is terminated and a new burst transaction is launched. Ongoing burst terminates at the next rising edge of **ifc\_launch\_clk** after the assertion of **IFCTA\_B**. The next burst's starting address is the last burst address + the number of bytes that needed to be transferred in the last burst.

$$\text{Address}_{n+1} = \text{Address}_n + (\text{number of bytes for current burst})$$

The next burst starts only after deassertion of **IFCTA\_B**. This is registered as an abort error in the status register and all transaction attributes are locked.

Since **IFCTA\_B** is synchronized through asynchronous FIFO, bus termination occurs only after a synchronization delay of **IFCTA\_B**. **IFCTA\_B** should be asserted for at least one **IFC\_CLK** cycle if it is synchronous (that is, meeting setup and hold time); otherwise, asynchronous **IFCTA\_B** should be asserted for a minimum of two **IFC\_CLK** cycles.



**Figure 25-48. Normal GPCM program operation - transaction aborted**

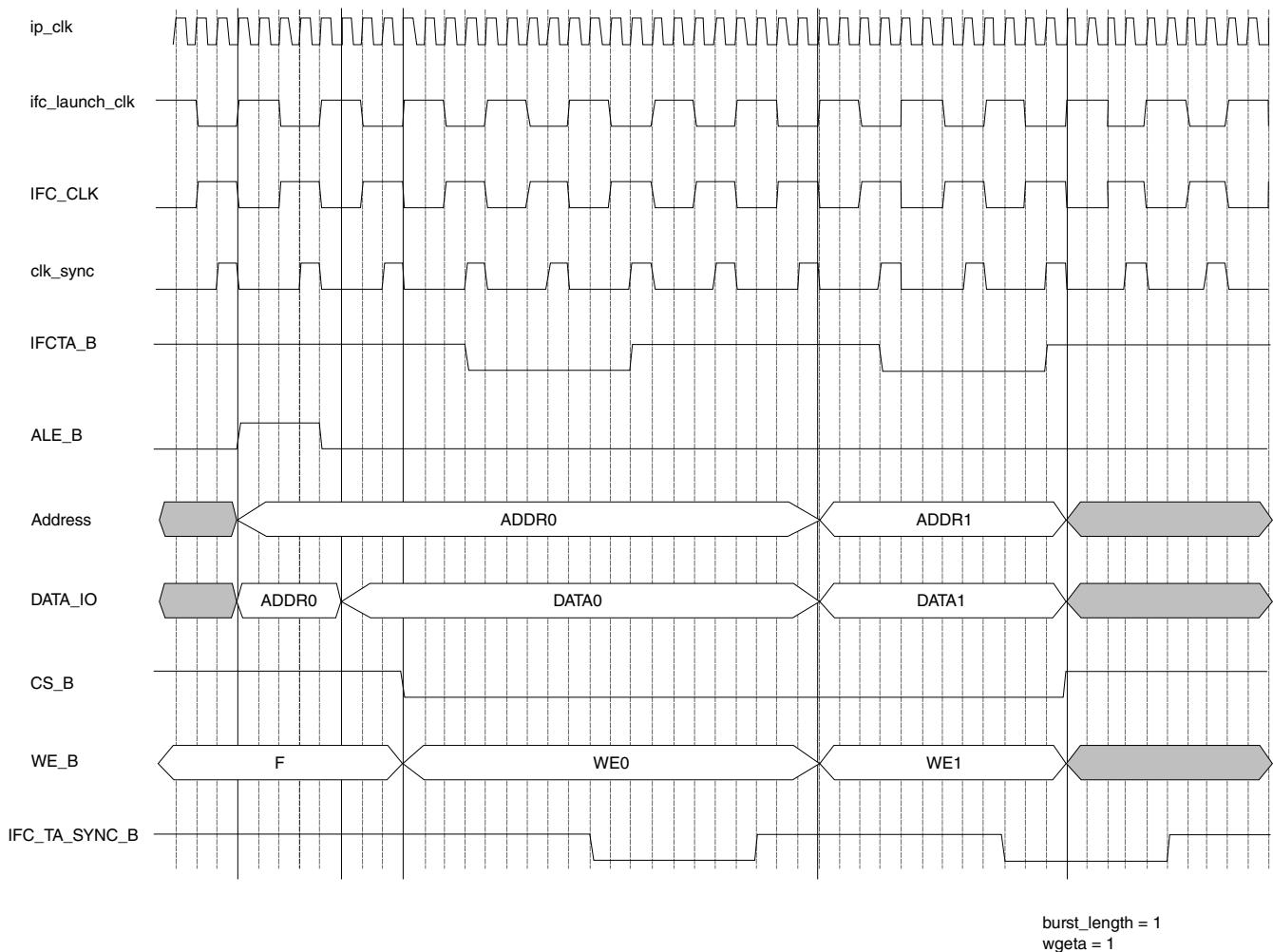
### 25.7.1.1.2 Normal GPCM external termination-based program operation

When WGETA is set, the next address and data are sent only after sampling the rising edge of IFCTA\_B or after timeout (as defined in register CSORn[GPTO]).

IFCTA\_B is synchronized to the IFC module input clock through asynchronous FIFO and only after the rising edge of IFCTA\_B, the new beat is sent over the interface. The next data and address beat is sent in sync with the positive edge of ifc\_launch\_clk. The last beat of the last burst completes at the next rising edge of ifc\_launch\_clk after assertion of IFCTA\_B, without waiting for deassertion of IFCTA\_B and CS\_B is deasserted.

If transaction size is greater than GPCM port size, the transaction will be split into multiple port size accesses. For non-burst mode, the last split is terminated at the rising edge of ifc\_launch\_clk after assertion of IFCTA\_B or after timeout. For other splits, IFC waits for IFC\_TA deassertion and not available for further transactions until IFC\_TA is deasserted.

As IFCTA\_B is synchronized through asynchronous FIFO, bus termination occurs only after the synchronization delay of the IFCTA\_B signal. IFCTA\_B should be asserted for at least one IFC\_CLK cycle if it is synchronous (that is, meeting setup and hold time); otherwise, if it is asynchronous, IFCTA\_B should be asserted for a minimum of two IFC\_CLK cycles.



**Figure 25-49. Normal GPCM program operation - acknowledgment mode**

## 25.7.1.2 Normal GPCM read operation

In normal GPCM, the read operation depends on the register field CSOR $n$ [RGETA]. The following options, through an internal counter or through an external device giving an access termination signal, are explained below.

### 25.7.1.2.1 Normal GPCM internal counter-based read operation

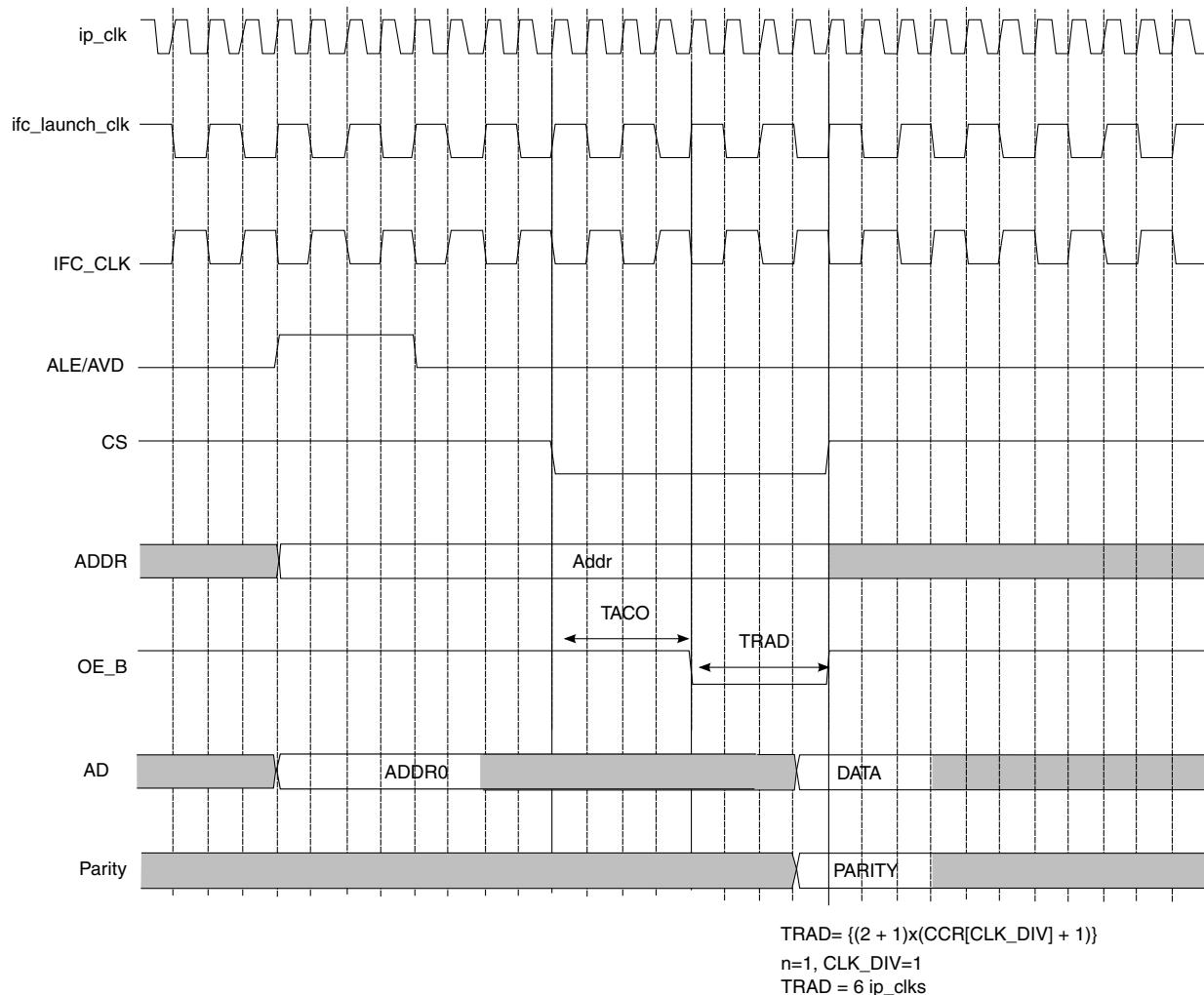
In this approach, read data from memory is sampled based on the programming of FTIM1\_CS $n$ \_GPCM[TRAD] and FTIM3\_CS $n$ \_GPCM[TAAD].

In non-burst mode, TRAD defines the pulse width of OE\_B. The read data is sampled on rising edge of IFC\_CLK just before the de-assertion of OE\_B

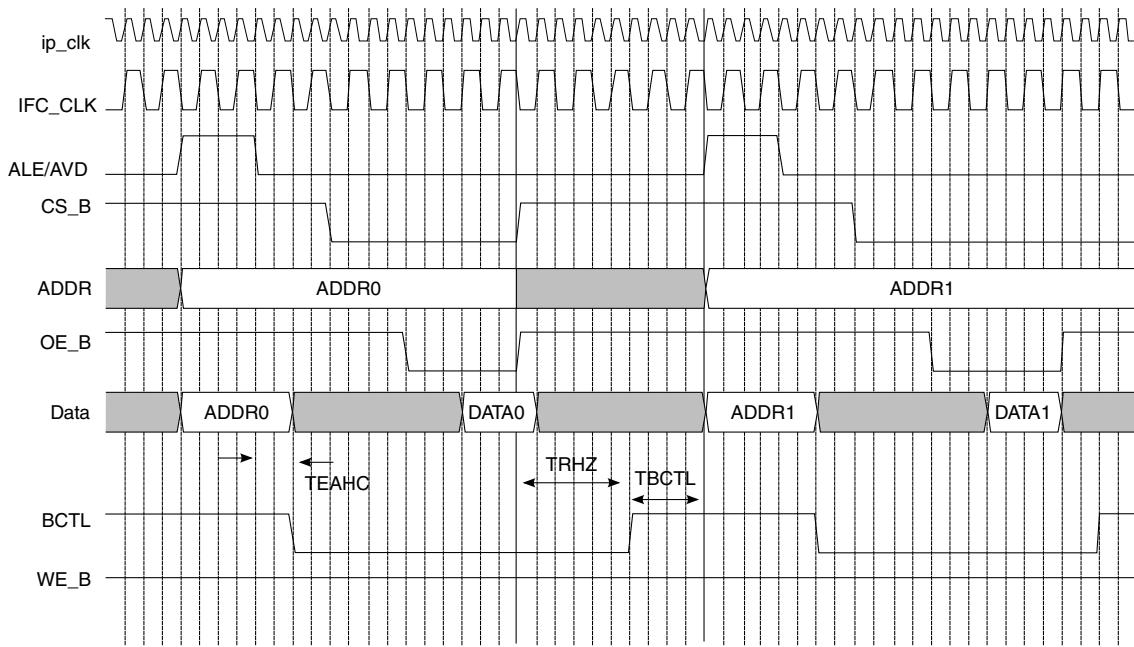
By default, (CCR[INV\_CLK\_EN]=1 and CCR[CLK\_DLY]=0) inverted clock of ifc\_launch\_clk goes out (IFC\_CLK) to the external device. Because the address is launched at the rising edge of ifc\_launch\_clk, the setup time on an external device for the address phase is half the cycle period of IFC\_CLK. The read data launched by the device will be sampled by the IFC at the next rising edge of IFC\_CLK. The programming of TRAD\_GPCM and TAAD\_GPCM should be as follows:

- For non-burst mode TRAD value should be programmed as  $\{(2 + n) * (CCR[CLK_DIV] + 1)\}$ , where n defines the memory access time in terms of IFC\_CLK. Memory takes n ifc\_clk to output data after output enable is asserted or after new address is sampled. n can take values 0,1,2,3...
- For burst mode TRAD/TAAD programmed will be  $\{(2 + n) * (CCR[CLK_DIV] + 1)\}$ , where n defines the memory access time in terms of IFC\_CLK. Memory takes n IFC\_CLK to output data after output enable is asserted or after new address is sampled. n can take values 0,1,2,3...

The following figures explain the above-mentioned read data sampling schemes:



**Figure 25-50. Normal GPCM read operation - non-burst mode**



TRHZ: Extended hold time for slow memories to disable their bus drivers  
TBCTL: Bus turn-around time

**Figure 25-51. Normal GPCM back-to-back read operation**

When RGETA is programmed to 0 and access is aborted by ifc\_ta, then the current burst is terminated and new burst transaction is launched. The ongoing burst is terminated at the next rising edge of ifc\_launch\_clk after assertion of ifc\_ta. The next burst starting address will be the last burst address + number of bytes that needed to be received in the last burst.

$$\text{Address}_{n+1} = \text{Address}_n + (\text{number bytes for current burst})$$

The next burst starts only after deassertion of ifc\_ta.

An abort error is registered in the status registers and all the transaction attributes are locked. Since ifc\_ta is synchronized through asynchronous FIFO, bus termination occurs only after synchronization delay of ifc\_ta signal. ifc\_ta should be asserted for minimum two IFC\_CLK cycles.

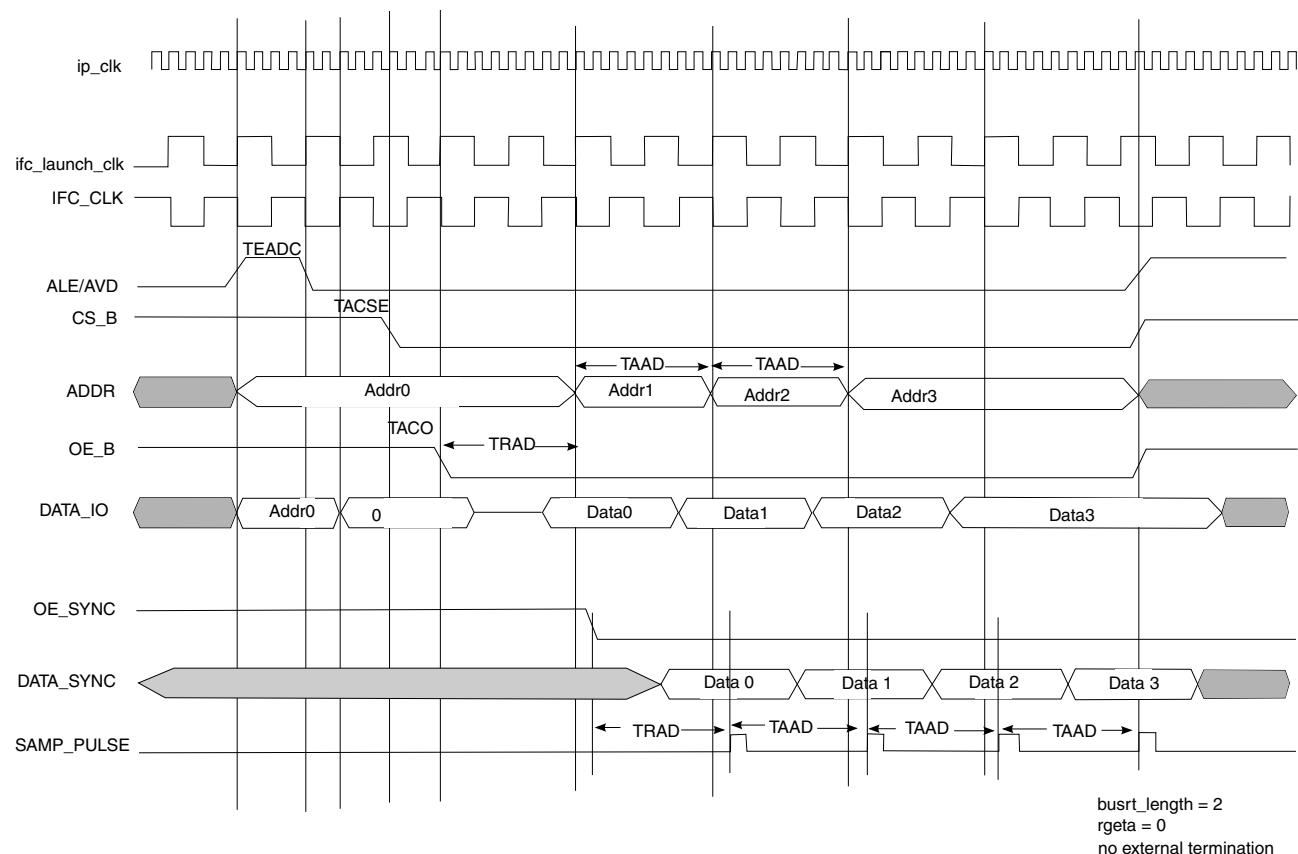
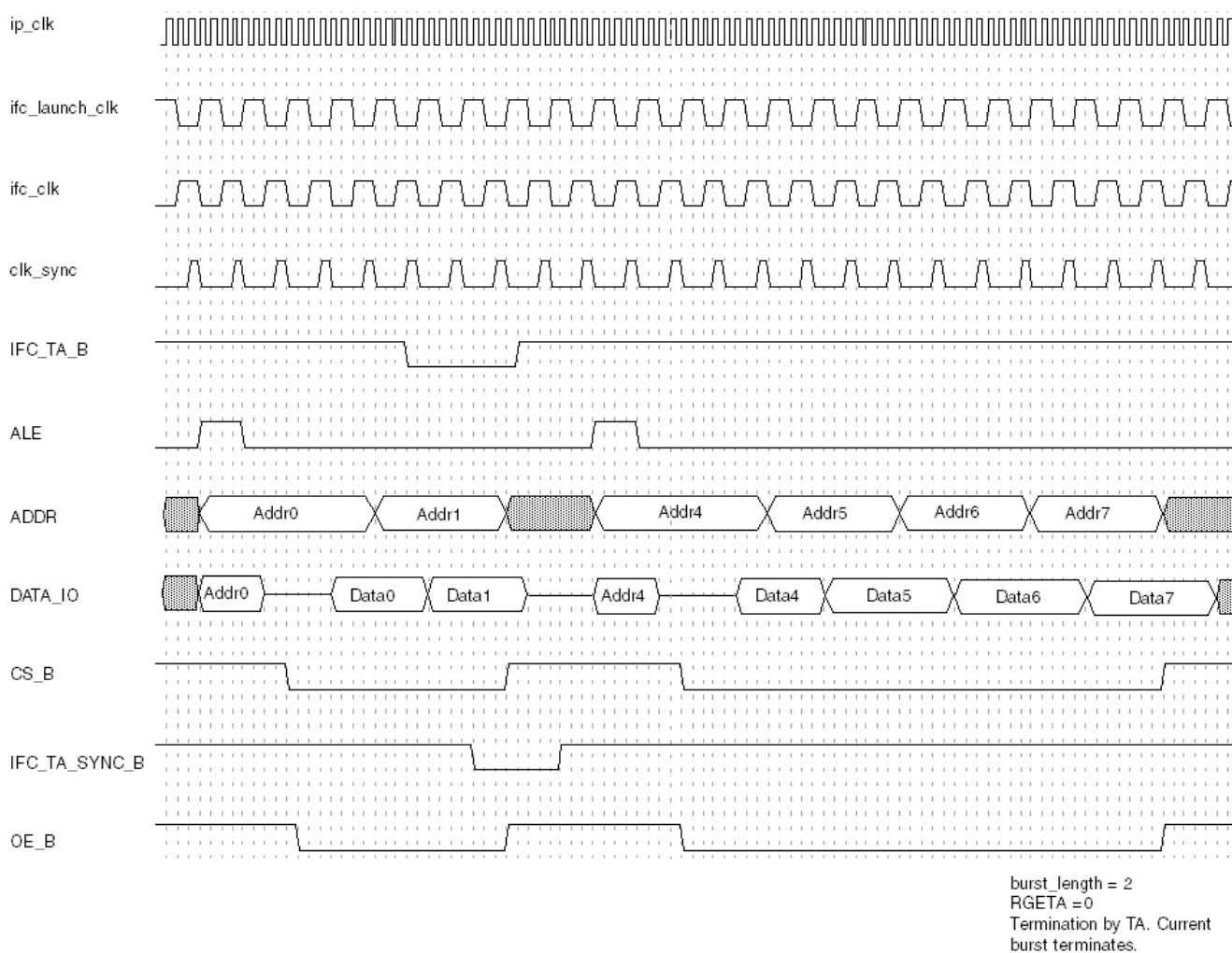


Figure 25-52. Normal GPCM read operation - burst mode

## General purpose chip-select machine (GPCM)



**Figure 25-53. Normal GPCM read operation - transaction aborted**

### 25.7.1.2.2 Normal GPCM external termination-based read operation

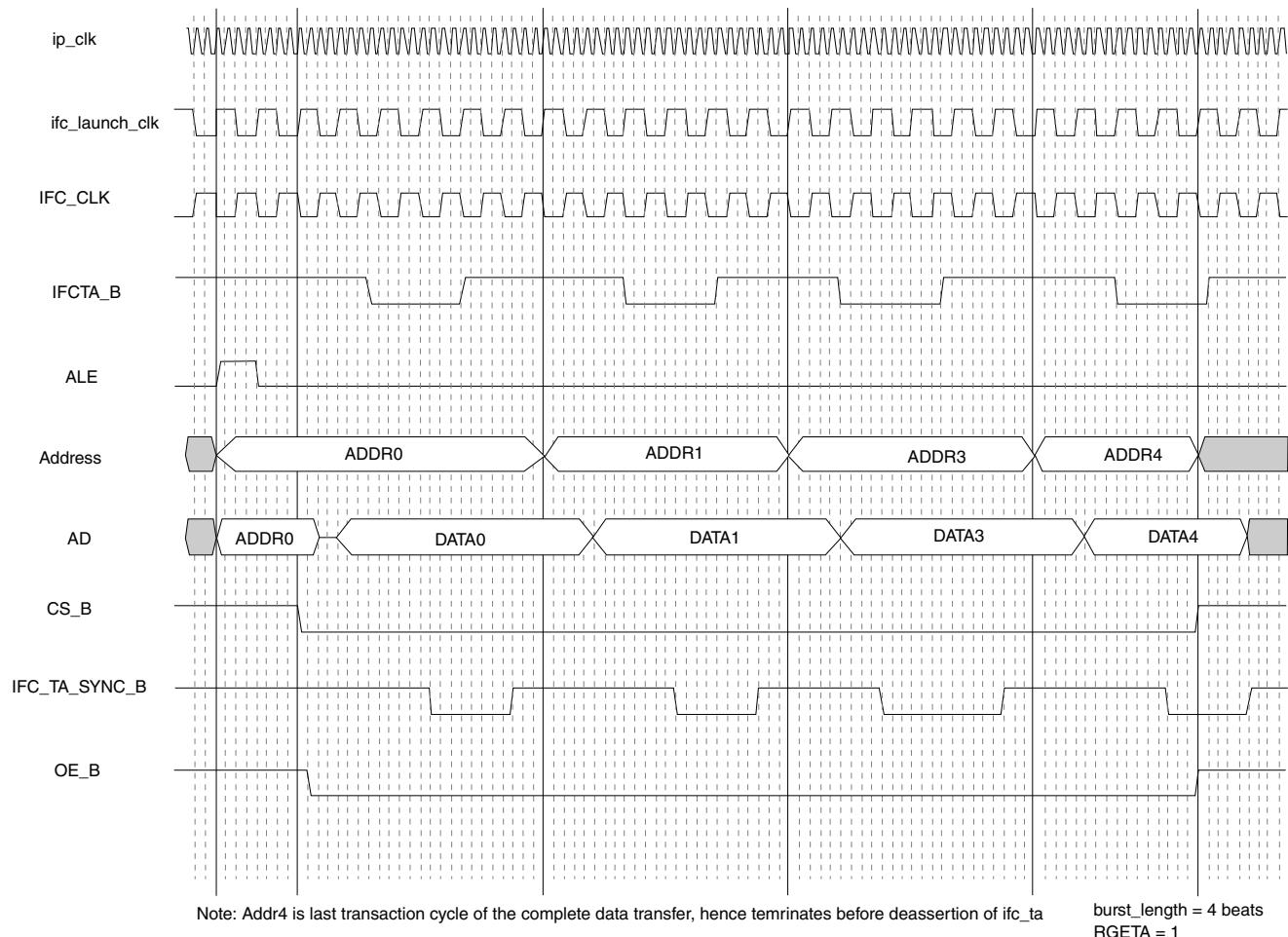
When RGETA is set to 1, each beat of the transaction waits for an external termination by IFCTA\_B or timeout.

For address transfer, the next address is sent only after IFCTA\_B is deasserted or after timeout. IFCTA\_B is synchronized to ip\_clk through the asynchronous FIFO and only after the rising edge new address beat is sent over the interface. The new address beat is sent in sync with ifc\_launch\_clk. The last transaction cycle does not wait for deassertion of IFCTA\_B; it ends at the next rising edge of ifc\_launch\_clk after assertion of IFCTA\_B or after timeout.

For read data sampling, the read data is sampled on the falling edge of IFCTA\_B. IFCTA\_B and data both are synchronized through the asynchronous FIFO.

For non-burst mode, access is terminated after IFCTA\_B is asserted or timeout.

In case of a timeout, an error is registered in the GPCM status register and all transaction attributes are locked. As IFCTA\_B is synchronized through the asynchronous FIFO, bus termination occurs only after a synchronization delay of the IFCTA\_B signal. IFCTA\_B should be asserted for at least one IFC\_CLK cycle if it is synchronous (that is, meeting setup and hold time); otherwise, if asynchronous, IFCTA\_B should be asserted for a minimum of two IFC\_CLK cycles.



**Figure 25-54. Normal GPCM read operation - acknowledgment mode**

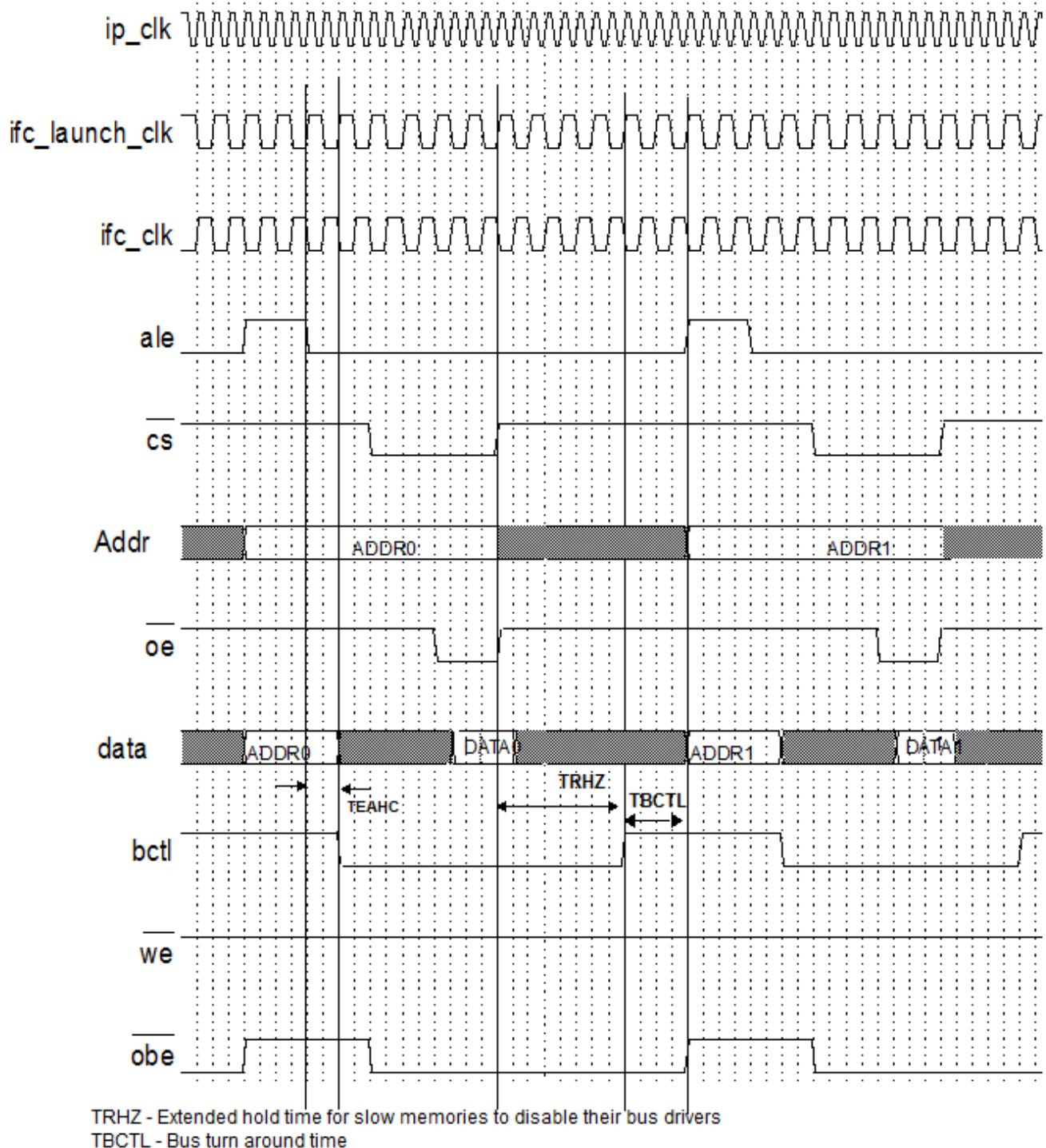
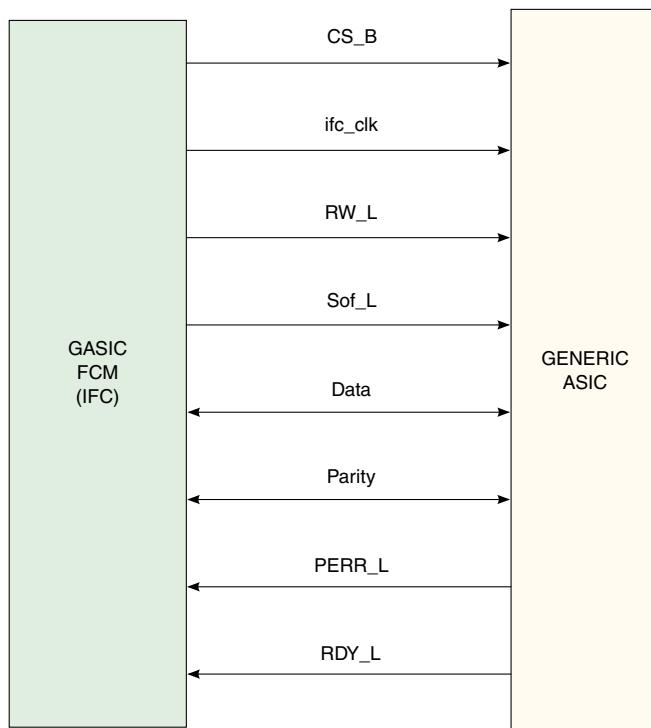


Figure 25-55. Normal GPCM back-to-back Read Operation

## 25.7.2 Generic ASIC mode of operation

This figure explains the interface of the IFC to a device connected through generic ASIC mode of GPCM (CSORn[GPMODE]=1).



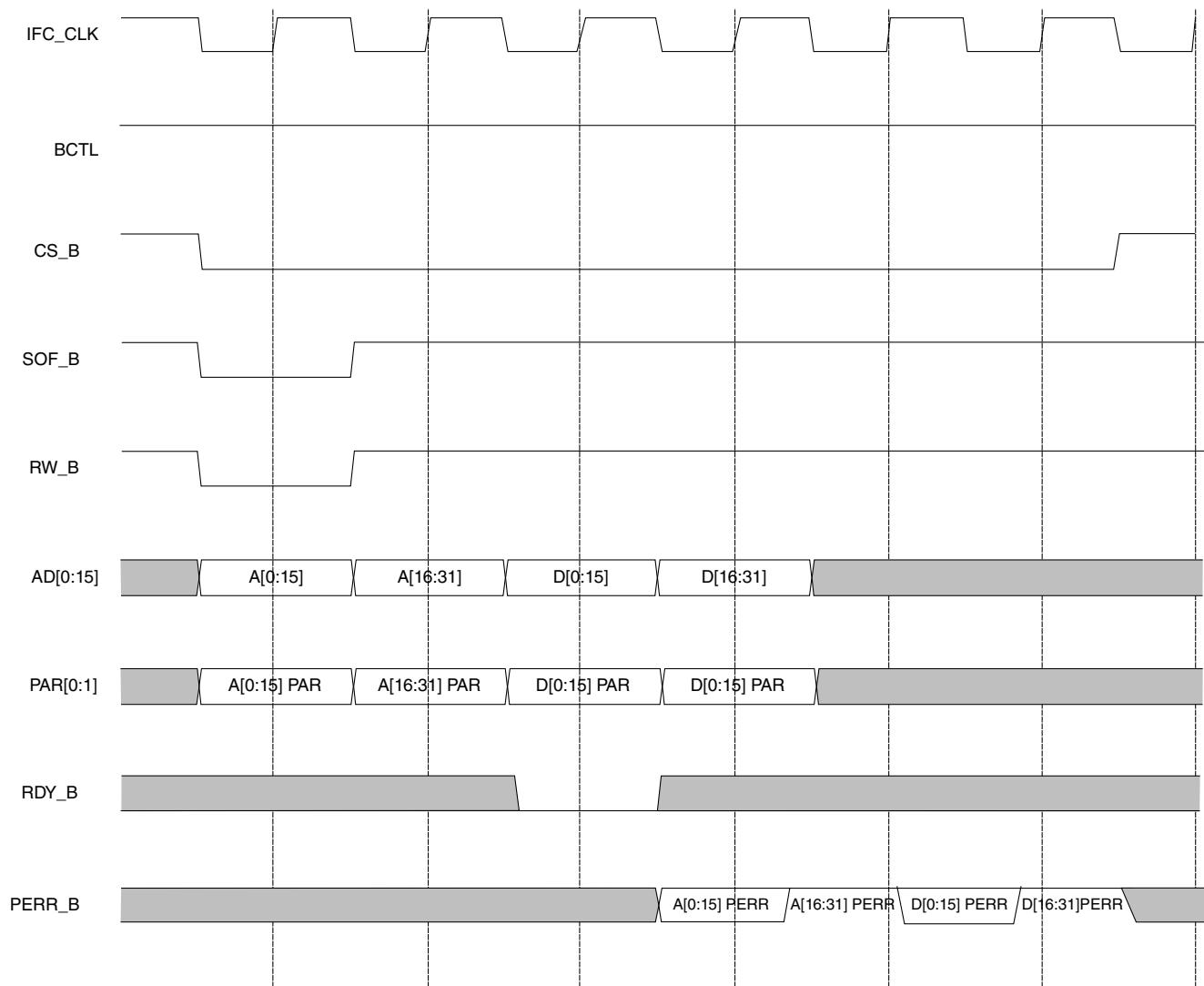
**Figure 25-56. Generic ASIC interface**

### 25.7.2.1 General ASIC program operation

Before initiating any general ASIC (GASIC) operation, the chip-select (CS\_B) is asserted to select the appropriate device.

The assertion of start of frame (SOF\_B) along with the low value on RW\_B indicates the start of address phase for write transfer. In the case of a 16-bit device, the upper 16 bits of address is driven on the AD[0:15] first, followed by the lower 16 bits on the next clock. After the address phase, the write data sequences are driven on AD lines based on the port size. Along with the address and write data phases, per-byte parity is driven on the parity lines. On the completion of write data phase, the host controller waits for the ready status (RDY\_B) and the parity error sampling (based on CSORn[GAPERRD] time) before deasserting the chip-select to complete the transfer. In the case where the device is not asserting RDY\_B before timeout (as defined in CSORn[GPTO]) occurs, the host terminates the transfer by deasserting the CS\_B.

## General purpose chip-select machine (GPCM)



**Figure 25-57. GASIC program operation with 16-bit device and zero-wait state**

For program operation, wait state represents delay (in terms of IFC\_CLK) between first write data beat versus RDY\_B assertion time.

For read, wait state represents delay (in terms of IFC\_CLK) between last address beat sent by controller and RDY\_L assertion time. If RDY\_L is asserted in next cycles of address sampled by GASIC device then it is termed as zero wait state.

### 25.7.2.2 Generic ASIC read data sampling

After selecting the general ASIC (GASIC) device, the host asserts start of frame (SOF\_B) and drive high value on RW\_B to indicate the start of address phase for read transfer.

The host drives the AD lines and the corresponding PAR parity lines in this phase to send the 16-bit of address to the device. After completing the address phase, the host stops driving the AD lines and waits for the assertion of ready status (RDY\_B) from the device. The assertion of RDY\_B indicates the start of read data phase. In the case the device has not asserted RDY\_B before timeout occurs, the host terminates the transfer by deasserting the CS\_B.

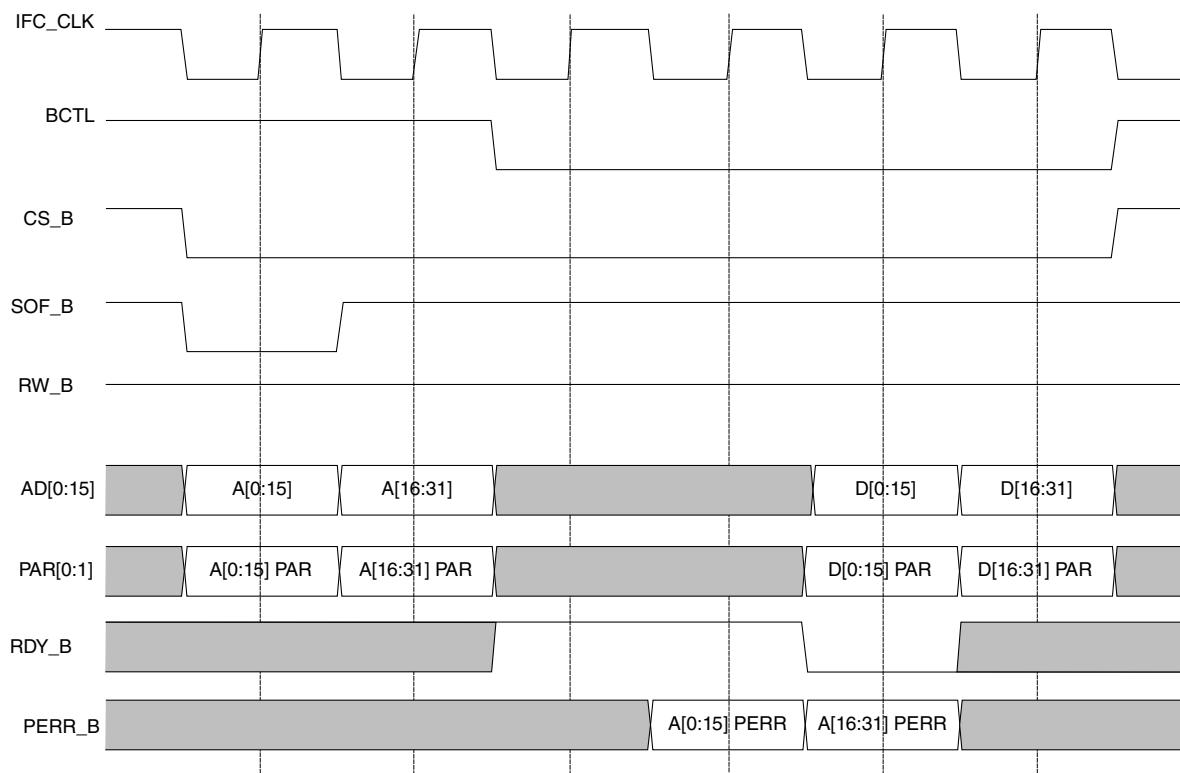
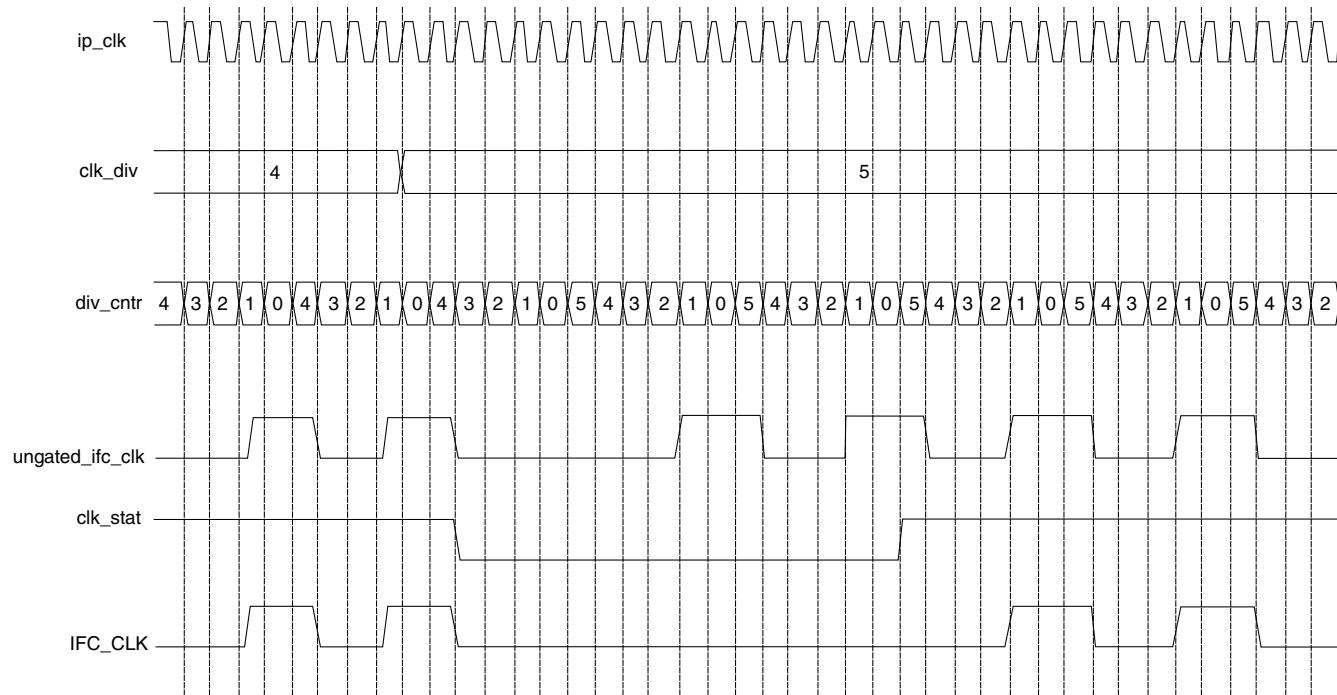


Figure 25-58. GASIC read operation with 16-bit device and 2-wait state

## 25.8 Clock generation module

The clock generation module generates a divided clock derived from the IFC module input clock depending on the programmed value in the CCR register.

Whenever there is a change in any clock division ratio, the clock is gated for few IFC module input clocks. The clock status is reported as unstable in the CSR register.



**Figure 25-59. External clock generation**

## 25.9 Initialization/Application information

The IFC can be used with separate address and data buses or with a multiplexed address/data bus.

This section provides guidelines for interfacing peripherals in the IFC's various modes.

### 25.9.1 Switching Interfaces in ONFi NAND

### 25.9.1.1 Activating the NVDDR Interface

After coming out of reset and performing device initialization, by default the asynchronous interface of the NAND device is activated. To switch to the NV-DDR interface, the following steps should be performed:

- Wait for OPC, if any previous operation is being executed on the NAND device.
- Program a FIR sequence (CMD0 - UA - WBCD - WFR - NOOP) that will issue the SET FEATURE command, followed by Wait For Ready, followed by NOOP.
- Issue the SET FEATURES (EFh) command. (CMD0)
- Write address 01h, which selects the timing mode. (UA)
- Write P1 with 1Xh for NVDDR devices, where "X" is the timing mode used in the synchronous interface (WBCD)
- Write P2-P4 as 00h-00h-00h. (WBCD)
- tWB time after the last data of the SET FEATURE command has been written the device goes into busy state (R/B\_B is pulled low). After tITC time elapses, the device enters ready state (R/B\_B transitions to high) and the NAND will now be in desired mode. The user is required to poll for the OPC status bit before issuing any new operation on the NAND device.
- Program the CSORn[NAND\_MODE] field to "0x01" for NV-DDR mode .
- If the user wants to use auto timing parameters, write the desired timing frequency value with which the NAND device is configured, to the CSOR\_EXTn[MODE\_FREQ] field. This will configure the FTIM registers, corresponding to the chip select whose MODE\_FREQ field has been written to, for the source synchronous device. The values programmed in the FTIM registers will be calculated based on the timing mode value programmed in CSOR\_EXTn[MODE\_FREQ], the clock divider value in the DDR-CCR register and the maximum permissible operating frequency (as per the ONFI ) for the programmed timing mode. Also the CSOR\_EXTn[AUTO\_TIM\_PARAMS\_SEL] should be set.
- If the user wants to use the FTIM registers to program timing parameters, the AUTO\_TIM\_PARAMS\_SEL bit should not be set.
- Before starting read operation using NVDDRmode, user must wait for DLL to get lock as specified in [DLL configuration guideline \(valid when IFC is in NVDDR Mode\)](#).

### 25.9.1.2 Switching to the asynchronous interface

To activate the asynchronous NAND interface, once the NVDDR interface is active, program the NAND\_CSORn[NAND\_MODE] to set the asynchronous mode.

The user should ensure that there is no activity on the NAND interface while programming this register field. Then perform the following steps:

- Wait for OPC, if any previous operation is being executed on the NAND device.
- Program the CSORn[NAND\_MODE] field to 00hh.
- Program FIR sequence (CMD0 -WFR - NOOP) that will issue the reset command (FFh) to the NAND device using an asynchronous command cycle.  $t_{WB}$  time after the command is issued, the RB\_B signal goes low indication the device is in busy state. After  $t_{RST}$  time, the device will be in ready state. The user should poll for the OPC status bit before issuing any new operations on the NAND device.

Once the above FIR sequence is complete, the NAND device will be in asynchronous mode and will be set to timing mode 0.

### **25.9.1.3 Switching timing modes when configured in NVDDR mode**

To switch timing modes on the device (with NVDDR interface active) the following steps should be performed:

- Wait for OPC, if any previous operation is being executed on the NAND device.
- Program a FIR sequence (CMD0 - UA - WBCD - WFR - NOOP) that will issue the SET FEATURE command, followed by Wait For Ready, followed by NOOP.
  - Issue the SET FEATURES (EFh) command. (CMD0)
  - Write address 01h, which selects the timing mode. (UA)
  - Write P1 with 1Xh for NVDDR devices, where "X" is the timing mode used in the synchronous interface (WBCD)
  - Write P2-P4 as 00h-00h-00h. (WBCD)
- $t_{WB}$  time after the last data of the SET FEATURE command has been written the device goes into busy state (R/B\_B is pulled low). After  $t_{ITC}$  time elapses, the device enters ready state (R/B\_B transitions to high) and the NAND will now be in desired mode. The user is required to poll for the OPC status bit before issuing any new operation on the NAND device.
- If the user wants to use auto timing parameters, write the desired timing frequency value with which the NAND device is configured, to the CSOR\_EXTn[MODE\_FREQ] field. This will configure the FTIM registers, corresponding to the chip select whose MODE\_FREQ field has been written to, for the source synchronous device. The values used for interface signal timing generation will be calculated based on the timing mode value programmed in CSOR\_EXTn[MODE\_FREQ], the clock divider value in the DDR-CCR register and the maximum permissible operating frequency (as per the ONFI spec) for the

- programmed timing mode. Also the CSOR\_EXTn[AUTO\_TIM\_PARAMS\_SEL] should be set
- If the user wants to use the FTIM registers to program timing parameters, the AUTO\_TIM\_PARAMS\_SEL bit should not be set.

## 25.9.2 ONFI mode timing parameters

For source-synchronous devices, there is an option to program the IFC with timing parameters as per the ONFI 2.2 timing parameters for all timing modes.

CSOR\_EXTn[ONFI\_MODE] is used to indicate the timing mode with which the chip is programmed and the CSOR\_EXTn[ONFI\_TIM\_PARAMS] is set to enable this feature. Based on the value of CSOR\_EXTn[ONFI\_MODE], the value of DDR\_CCR[DDR\_CLK\_DIV], and the maximum-permissible operating frequency for a particular timing mode (as per ONFI 2.2), the timing values are programmed in the number of IPG clock cycles.

The values of the timing parameters (rounded to the next higher integer) can be calculated as follows:

- ONFI\_TIMING\_PARAMETER + (MODE\_PERIOD/(2 x DDR\_CLK\_DIV + 2) - 1)
- MODE\_PERIOD/(2 x DDR\_CLK\_DIV + 2)

where

- ONFI\_TIMING\_PARAMETER is an ONFI timing parameter (such as tcad, tcs, and so on)
- MODE\_PERIOD is the minimum clock period (ns) for a particular ONFI mode.
- DDR\_CLK\_DIV is the clock division ratio programmed in the DDR\_CCR[DDR\_CLK\_DIV] register.

## 25.9.3 DLL configuration guideline (valid when IFC is in NVDDR Mode)

The DLL (NAND\_DLL\_LOW is) configured using the Configuration 0 and Configuration 1 registers. The following sequence should be followed before the DLL can be used to delay the incoming data strobe:

- The NAND\_DLL\_LOW is enabled when the DDR clock is enabled (by setting DDR CCR LOW [2]) and CSOR[NAND\_MODE] = NVDDR mode.
- The NAND\_DLL\_LOW is used for interface frequencies up to 133 MHz.

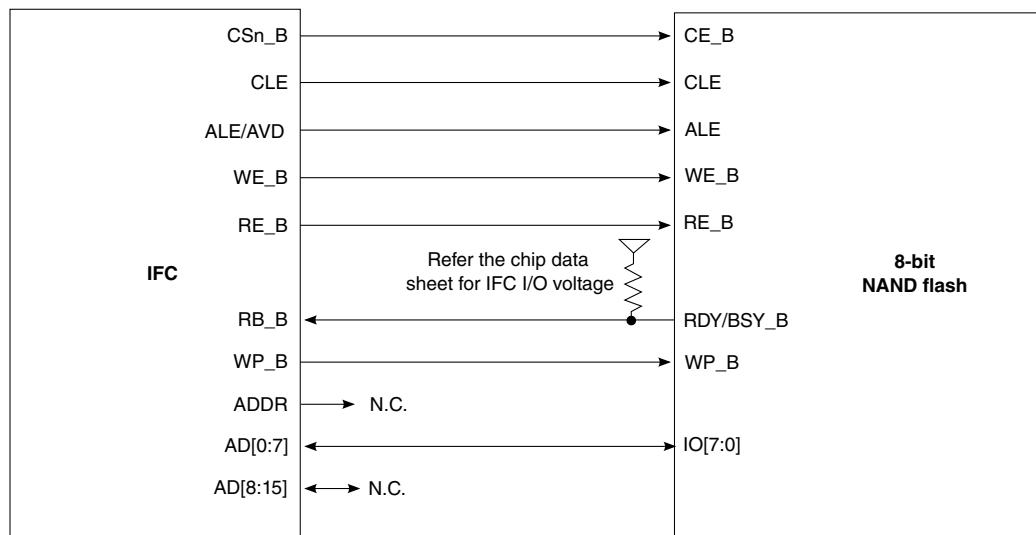
- The reset value of the dll\_slv\_dly\_target field in DLL Configuration 0 register for NAND\_DLL\_LOW is 0x7. This will ensure a delay of REF\_CLK/4 on the incoming DQS.
- The user must change the NAND\_DLL\_CFG1[DLL\_PD\_PULSE\_STRETCH\_SEL] bit from its default value 1'b1 to 1'b0 for reliable pulse detection and DLL lock.
- The user has to poll for the value of ‘1’ on the DLL\_STS\_SLV\_LOCK bit of NAND\_DLL\_LOW\_STAT (if enabled) register before issuing any access to the NAND Flash. Neglecting this would result in incorrect read data capture by IFC.
- If the CSOR\_EXT[MODE\_FREQ] bit is changed, then the delay chain needs to be reset to make it lock again.

## 25.9.4 IFC flash connections

### 25.9.4.1 NAND flash connections

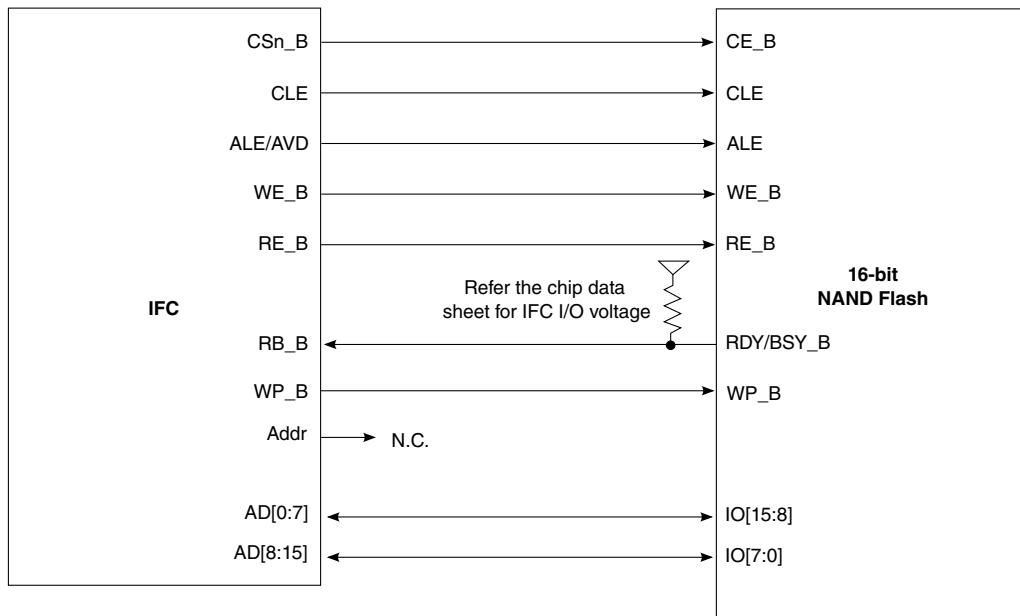
The NAND FCM provides a glueless interface to 8- or 16-bit parallel-bus NAND flash EEPROM devices.

This figure shows a simple connection between an 8-bit port size NAND flash EEPROM and the IFC. In NAND FCM mode, commands, address bytes, and data are all transferred on AD[0:7].



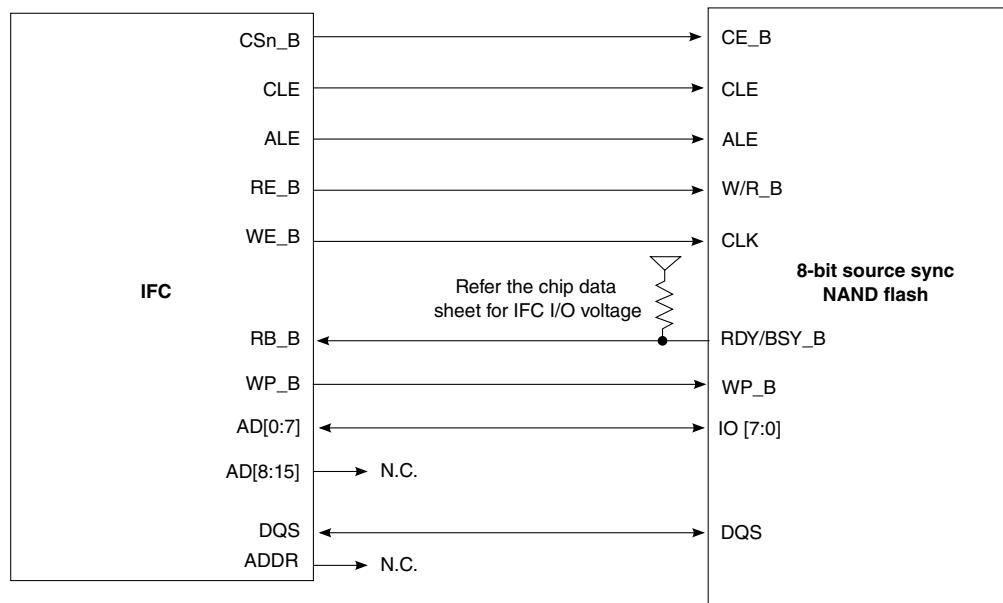
**Figure 25-60. IFC to 8-bit asynchronous NAND device interface**

This figure shows connection of a 16-bit NAND flash to IFC. Commands and address bytes appear on AD[8:15].



**Figure 25-61. IFC to 16-bit asynchronous NAND flash device interface**

This figure shows a simple connection between an 8-bit port size NAND flash EEPROM (in source synchronous mode) and the IFC. In NAND FCM mode, commands, address bytes, and data are all transferred on AD[0:7].



**Figure 25-62. IFC to 8-bit NVDDR NAND flash device interface**

### 25.9.4.1.1 Switching to the synchronous interface

After coming out of reset and performing device initialization, by default the asynchronous interface of the NAND device is activated.

To switch to the synchronous interface:

- Wait for OPC, if any previous operation is being executed on the NAND device.
- Program a FIR sequence (CMD0 -UA-WBCD-WFR - NOOP) that issues the SET FEATURE command, followed by wait-for-ready, followed by NOOP.
  - Issue the SET FEATURES (EFh) command. (CMD0)
  - Write address 01h, which selects the timing mode. (UA)
  - Write P1 with  $1nh$ , where " $n$ " is the timing mode used in the synchronous interface (WBCD).
  - Write P2-P4 as 00h-00h-00h. (WBCD)
- $t_{WB}$  time after the last data of the SET FEATURE command has been written the device goes into busy state (RB\_B is pulled low). After  $t_{ITC}$  time elapses, the device enters ready state (RB\_B transitions to high) and the NAND is now in source-synchronous mode. It is required to poll for the OPC status bit before issuing any new operation on the NAND device.
- Program the CSORn[NAND\_MODE] field to 01h.
- To use ONFI timing parameters, write the desired timing mode value (0-5) with which the NAND device is configured, to the CSOR\_EXTn[ONFI\_MODE] field. This configures the FTIM registers, corresponding to the chip-select whose ONFI\_MODE field has been written to, for the source-synchronous device. The values programmed in the FTIM registers are calculated based on the timing-mode value programmed in CSOR\_EXTn[ONFI\_MODE], the clock divider value in the DDR-CCR register, and the maximum permissible operating frequency (as per the ONFI 2.2 spec) for the programmed timing mode. Also the CSOR\_EXTn[ONFI\_TIM\_PARAMS] should be set.
- To use the FTIM registers to program timing parameters, do not set the ONFI\_TIM\_PARAMS bit.

### 25.9.4.1.2 Switching to the asynchronous interface

To activate the asynchronous NAND interface, once the NVDDR interface is active, program the NAND\_CSORn[NAND\_MODE] to set the asynchronous mode.

The user should ensure that there is no activity on the NAND interface while programming this register field. Then perform the following steps:

- Wait for OPC, if any previous operation is being executed on the NAND device.

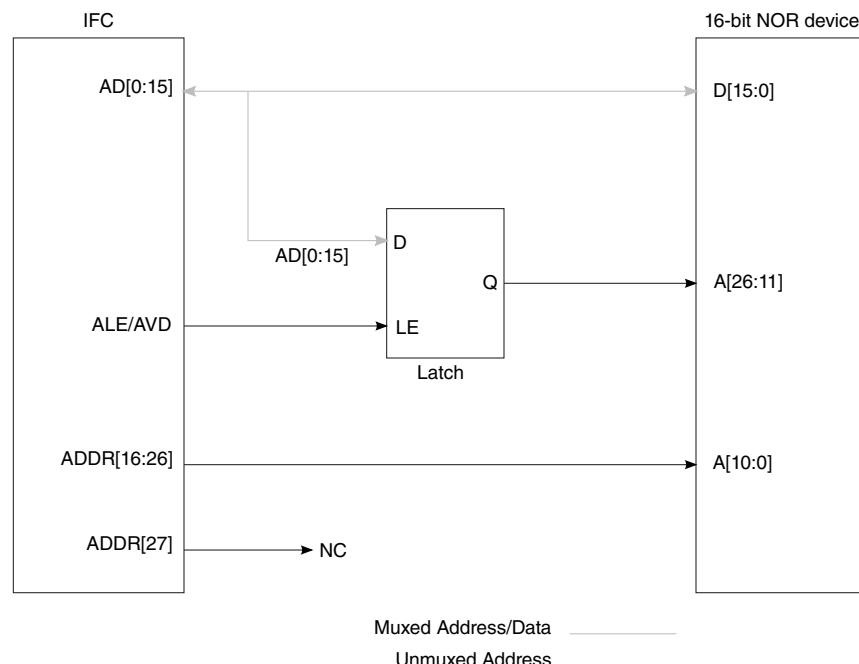
- Program the CSORn[NAND\_MODE] field to 00hh.
- Program FIR sequence (CMD0 -WFR - NOOP) that will issue the reset command (FFh) to the NAND device using an asynchronous command cycle.  $t_{WB}$  time after the command is issued, the RB\_B signal goes low indication the device is in busy state. After  $t_{RST}$  time, the device will be in ready state. The user should poll for the OPC status bit before issuing any new operations on the NAND device.

Once the above FIR sequence is complete, the NAND device will be in asynchronous mode and will be set to timing mode 0.

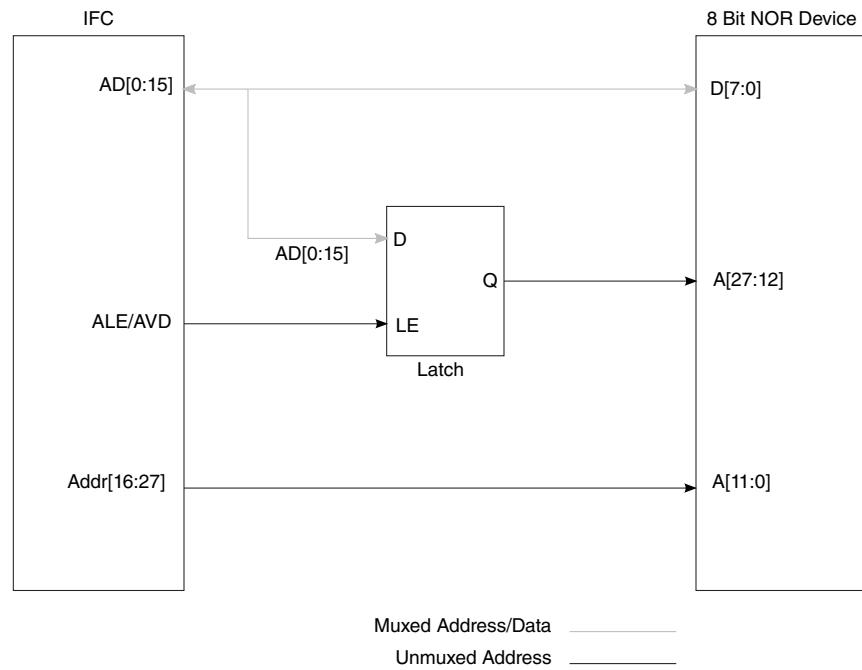
#### 25.9.4.2 NOR flash connections

For 16-bit devices, ADDR[26] is used and ADDR[27] is irrelevant; however, for 8-bit devices, ADDR[26:27] are necessary. If the bus width is 2 bytes wide, then ADDR[27] is a don't care and would not be connected.

This figure shows the IFC-to-NOR device interface.



**Figure 25-63. IFC to 16-bit NOR device interface**

**Figure 25-64. IFC to 8-bit NOR device interface**

## 25.9.5 Bus turnaround

Because the IFC uses multiplexed address and data, special consideration is given to avoid bus contention at bus turnaround.

The following cases must be examined:

- Address phase after previous read
- Read-data phase after address phase
- Read-modify-write cycle for parity-protected memory banks

The bus does not change direction for the following cases, so no special attention is required:

- Continued burst after the first beat
- Write-data phase after address phase
- Address phase after previous write

### 25.9.5.1 Address phase after previous read

During a read cycle, the memory/peripheral drives the bus and the bus transceiver drives AD.

After the data has been sampled, the output drivers of the external device must be disabled, which can take some time.

After the previous cycle ends, BCTL goes high and changes the direction of the bus transceiver. The IFC then inserts a bus turnaround time (that is, TBCTL) to avoid contention. The external device has now already placed its data signals in high impedance and no bus contention occurs.

### 25.9.5.2 Read-data phase after address phase

During the address phase, AD actively drives the address and BCTL is high, driving the bus transceivers in the same direction as during a write.

After the end of the address phase, BCTL goes low and changes the direction of the bus transceiver.

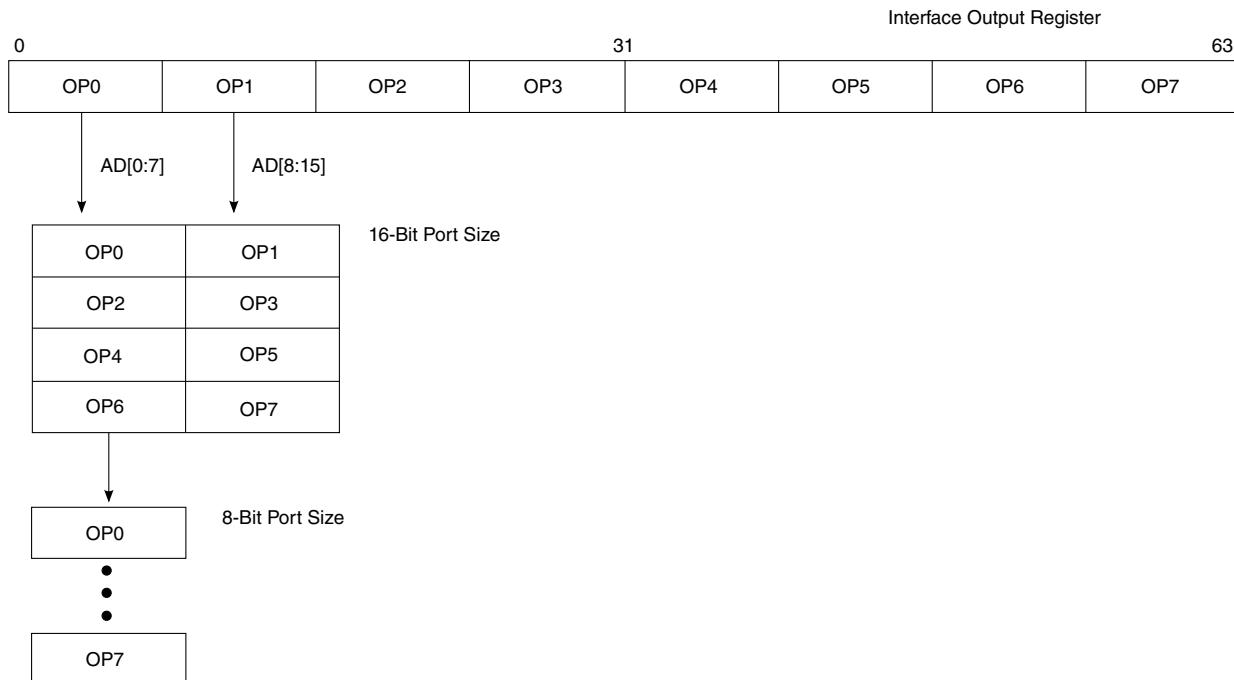
### 25.9.5.3 Read-modify-write cycle for parity protected memory banks

Principally, a read-modify-write cycle is a read cycle immediately followed by a write cycle. Because the write cycle has a new address phase in any case, this essentially is the same as an address phase after a previous read.

## 25.9.6 Interfacing to different port sizes

The IFC supports 8- 16-bit data port sizes.

However, the IFC requires that the portion of the data bus used for a transfer to or from a particular port size be fixed. A 16-bit port must reside on AD[0:15], and an 8-bit port on AD[0:7]. This figure shows the device connections on the data bus.



**Figure 25-65. Interface to different port-size devices**

## 25.9.7 Command sequence examples for NAND flash EEPROM

To program the IFC and FCM for executing NAND flash command sequences, obtain command codes and pause states from the relevant NAND flash device datasheet and program into FCM configuration registers.

This section describes some common sequences for multi-gigabit NAND flash EEPROMs; however, details should be verified against manufacturer's specific programming data.

Throughout these examples, it is assumed that one or more banks of the IFC have been configured under FCM control with base address, port size, ECC mode, and timing parameters configured in accordance with the device's data sheet .

### 25.9.7.1 NAND flash soft reset command sequence example

An example of configuring FCM to execute a soft reset command to a 2 KB-page NAND flash is shown in this table.

This sequence does not require the use of the shared FCM buffer RAM. At the conclusion of the sequence, IFC issues a command complete interrupt if interrupts are enabled.

**Table 25-12. FCM register settings for soft reset**

Register	Initial contents	Description
NAND_FCR0	FF00_0000h	CMD0 = FFh = reset command; other commands unused
ROW0	-	Unused
COL0	-	Unused
NAND_BC	-	Unused
NAND_FSR	-	Unused
NAND_FIR0, NAND_FIR1, NAND_FIR2	2400_0000h, 0000_0000h, 0000_0000h	OP0 = CMD0 = command 0; OP1-OP14 = NOOP

### 25.9.7.2 NAND flash read status command sequence example

An example of configuring FCM to execute a status read command to 2 KB-page NAND flash is shown in this table.

This sequence does not require the use of the shared FCM buffer RAM, but reads the NAND flash status into NAND\_FSR.

At the conclusion of the sequence, the IFC issues a command complete interrupt if interrupts are enabled.

**Table 25-13. FCM register settings for status read**

Register	Initial contents	Description
NAND_FCR0	7000_0000h	CMD0 = 70h = read status command; other commands unused
ROW0	-	Unused
COL0	-	Unused
NAND_BC	-	Unused
NAND_FSR	-	Status returned in RS0
NAND_FIR0, NAND_FIR1, NAND_FIR2	25C0_0000h, 0000_0000h, 0000_0000h	OP0 = CMD0 = command 0; OP1 = RDSTAT = read status to NAND_FSR; OP2-OP14 = NOOP

### 25.9.7.3 NAND flash read identification command sequence example

An example of configuring FCM to execute a status ID command to 2 KB-page NAND flash is shown in this table.

This sequence uses the shared FCM buffer RAM to receive the bytes of ID during the sequence.

At the conclusion of the sequence, IFC issues a command complete interrupt if interrupts are enabled.

**Table 25-14. FCM register settings for ID read**

Register	Initial contents	Description
NAND_FCR0	9000_0000h	CMD0 = 90h Read ID command; other commands unused
ROW3	Row address (0000_0020h)	Locates the block and the page within that block to be accessed
COL0	0000_0000h	Locates the byte to be accessed within a given page MS = 0
NAND_BC	0000_0004h	BC = 4 to read the ONFI Signature (for ONFI devices)
NAND_FSR	-	Unused
NAND_FIR0, NAND_FIR1, NAND_FIR2	2608_4000h, 0000_0000h, 0000_0000h	OP0 = CMD0 = command 0; OP1 = UA = Send row address programmed in row address register 3. Use SRAM buffer 0 to store the read ID bytes; OP2 = RB = Read BC bytes of data into the SRAM buffer 0; OP3-OP14 = NOOP

The above sequence uses opcode UA to send the address stored in row address register 3. This is done so that the data transfer, that is, the read data is stored in buffer 0 of the internal SRAM. Thus, the UA could also be used for operations, such as read page parameter, get feature, set feature, read unique ID always using buffer 0 of the internal SRAM.

### 25.9.7.4 NAND flash page read command sequence example

An example of configuring FCM to execute a random page read command to 2-Kbytes page NAND flash is shown in the table below.

This sequence reads an entire page (main and spare region) into the shared FCM buffer RAM, checking ECC (if enabled) as it proceeds.

At the conclusion of the sequence, IFC will issue a command complete interrupt if interrupts are enabled. Once the sequence has completed, the shared buffer (buffer 1 for page index 5) and transfer error registers are valid.

**Table 25-15. FCM register settings for page read**

Register	Initial contents	Description
NAND_FCR0	0030_0000h	CMD0 = 00h = random read address entry; CMD1 = 30h = read page
ROW0	row address (for example, 0000_0005h locates page5 in block0)	Locates the block and the page within that block to be accessed
COL0	0000_0000h	Locates the byte to be accessed within a given page MS = 0
NAND_BC	0000_0000h	BC = 0 to read entire 2112-byte page
NAND_FSR	-	unused
NAND_FIR0, NAND_FIR1,NAN D_FIR2	2411_4A68h, 0000_0000h, 0000_0000h	OP0 = CMD0 = command 0; OP1 = CA0 = column address; OP2 = RA0 = page address; OP3 = CMD1 = command 1; OP4 = RBCD = read BC bytes of data into FCM buffer; OP5 - OP14 = NOOP

### 25.9.7.5 NAND flash program command sequence example

An example of configuring FCM to execute a program command to 2 KB-page NAND flash is shown in this table.

This sequence writes an entire page (main and spare region) from the shared FCM buffer RAM, generating ECC (if enabled) as it proceeds.

The shared buffer (buffer 1 for page index 5) must be initialized by software prior to starting the sequence. At the conclusion of the sequence, the IFC issues a command complete interrupt if interrupts are enabled. The status of the programming operation is returned in NAND\_FSR.

Note that operations specified by OP5 and OP6 (status read) should never be skipped while programming a NAND flash device, because it may happen that a new command is issued to the NAND flash device even when the device has not yet finished processing the previous request. This may result in unpredictable behavior.

**Table 25-16. FCM register settings for page program**

Register	Initial contents	Description
NAND_FCR0	8070_1000h	CMD0 = 80h = page address and data entry; CMD1 = 70h = read status CMD2 = 10h = program page;

*Table continues on the next page...*

**Table 25-16. FCM register settings for page program (continued)**

Register	Initial contents	Description
ROW0	Row address (for example, 0000_0005h locates page5 in block0)	Locates the block and the page within that block to be accessed
COL0	0000_0000h	Locates the byte to be accessed within a given page MS = 0
NAND_BC	0000_0000h	BC = 0 to read entire 2112-byte page
NAND_FSR	-	Returns with AS0 holding program status
NAND_FIR0, NAND_FIR1, NAND_FIR2	2411_592Ch, 49C0_0000h, 0000_0000h	OP0 = CMD0 = command 0; OP1 = CA0 = column address; OP2 = RA0 = page address; OP3 = WBCD = write BC bytes of data from buffer; OP4 = CMD2 = command 2; OP5 = CW1 = wait on flash ready and issue command 1; OP6 = RDSTAT = read erase status into NAND_FSR; OP7-OP14 = NOOP

### 25.9.7.6 Read status command during busy period of program/erase operation

During program and erase operation, the device permits the user to read status during its busy period.

To achieve this, use the sequence of commands in this table.

**Table 25-17. FCM register settings for read status during program busy period**

Register	Initial contents	Description
NAND_FCR0	8070_1000h	CMD0 = 80h = page address and data entry; CMD1 = 0x70 = read status CMD2 = 10h = program page;
ROW0	row address (for example, 0000_0005h locates page5 in block0)	Locates the block and the page within that block to be accessed
COL0	0000_0000h	Locates the byte to be accessed within a given page MS = 0
NAND_BC	0000_0000h	BC = 0 to read entire 2112-byte page
NAND_FSR	-	Returns with AS0 holding program status
NAND_FIR0, NAND_FIR1, NAND_FIR2	2411_592Ch, 49C0_0000h, 00000_000h	OP0 = CMD0 = command 0; OP1 = CA0 = column address; OP2 = RA0 = page address; OP3 = WBCD = write BC bytes of data from buffer; OP4 = CMD2 = command 2; OP5 = NWAIT = program NCFGR[NUM_WAIT] for tWB time.

**Table 25-17. FCM register settings for read status during program busy period**

Register	Initial contents	Description
		OP6 = CMD1 = command 1; OP7 = RDSTAT = read erase status into NAND_FSR; OP8–OP14 = NOOP

In this sequence, use opcode (NWAIT, CMD1) instead of CW1 to issue a read status during the device-busy phase. This is different from the sequence [NAND flash program command sequence example](#), where a read status is issued after the device becomes ready.

### 25.9.7.7 Valid opcode transitions in the IFC

This table describes the valid opcode transitions.

Ensure that the FIR sequence is programmed using only valid opcode transitions.

**Table 25-18. Supported opcode transitions (in FIR)**

Current opcode	Next opcode supported <i>n=0-7, m=0-3</i>								
	CAm/ RAM	CMDn	UA	RBCD/ BTRD/ SBRD/ RDSTAT/ RB_B	CWn	WFR	NWAIT	NOOP	
CMDn	CAm/ RAM	CMDn	UA	RDSTAT/ RB	CWn	WFR	NWAIT	NOOP	
CAm	CAm/ RAM	CMDn	UA	WBCD	RDSTAT/R B	CWn	WFR	NWAIT	NOOP
RAM	RAM	CMDn	UA	WBCD	RDSTAT/R B	CWn	WFR	NWAIT	NOOP
UAm	RAM	CMDn	UA	WBCD	RDSTAT/R B	CWn	WFR	NWAIT	NOOP
WBCD	CMDn	CWn	WFR	NWAIT	NOOP				
CWn	CAm/ RAM	CMDn	UA	RBCD/ BTRD/ SBRD/ RDSTAT/ RB_B	CWn	WFR	NWAIT	NOOP	
WFR	CMDn	RBCD/ SBRD	NWAIT	NOOP	-	-	-	-	
NWAIT	CMDn	CAm/ RAM	UA	WBCD	RDSTAT/R B	CWn	WFR	NOOP	
RBCD/ BTRD/ SBRD/ RDSTAT/ RB	RDSTAT	CMDn	CWn	CAm/ RAM	NWAIT	NOOP	-	-	



# **Chapter 26**

## **Inter-Integrated Circuit (I2C)**

### **26.1 The I2C module as implemented on the chip**

This section provides details about how the I2C module is implemented on the chip.

#### **26.1.1 LS1043A I2C module integration**

The following table describes the I2C module integration into the chip:

**Table 26-1. I2C module integration**

Module	Module Base address
I2C1	218_0000
I2C2	219_0000
I2C3	21A_0000
I2C4	21B_0000

Additionally, both modules have been integrated with identical parameters and connections.

The remainder of this chapter refers to a single I2C module. Notes are included to indicate variations for multiple instantiations.

#### **26.1.2 LS1043A I2C module special consideration**

The I2C module implements the following parameter settings in the chip:

**Table 26-2. LS1043A I<sup>2</sup>C parameter settings**

I <sup>2</sup> C parameters	LS1043A parameter value
Stop mode support	Yes. Refers to LPM20 low power mode of the chip.

The table below provides the clock sources to each I<sup>2</sup>C module:

**Table 26-3. LS1043A I<sup>2</sup>C clocking**

Module	LS1043A clocking source
I <sup>2</sup> C1	platform clock
I <sup>2</sup> C2	platform clock
I <sup>2</sup> C3	platform clock
I <sup>2</sup> C4	platform clock

## 26.2 Overview

This chapter describes the Inter-Integrated Circuit (I<sup>2</sup>C) bus module implemented on this chip and presents the following topics:

- [Introduction to I<sup>2</sup>C](#)
- [External signal descriptions](#)
- [Memory map and register definition](#)
- [Functional description](#)
- [Initialization/application information](#)

## 26.3 Introduction to I<sup>2</sup>C

This section presents the following topics:

- [Definition: I<sup>2</sup>C module](#)
- [Advantages of the I<sup>2</sup>C bus](#)
- [Module block diagram](#)
- [Features](#)

- Modes of operation
- Definition: I<sup>2</sup>C conditions

### 26.3.1 Definition: I<sup>2</sup>C module

The I<sup>2</sup>C module is a functional unit that provides a two-wire— serial data (SDA) and serial clock (SCL) — bidirectional serial bus that provides a simple and efficient method of data exchange between this chip and other devices, such as microcontrollers, EEPROMs, real-time clock devices, analog-to-digital converters, and LCDs.

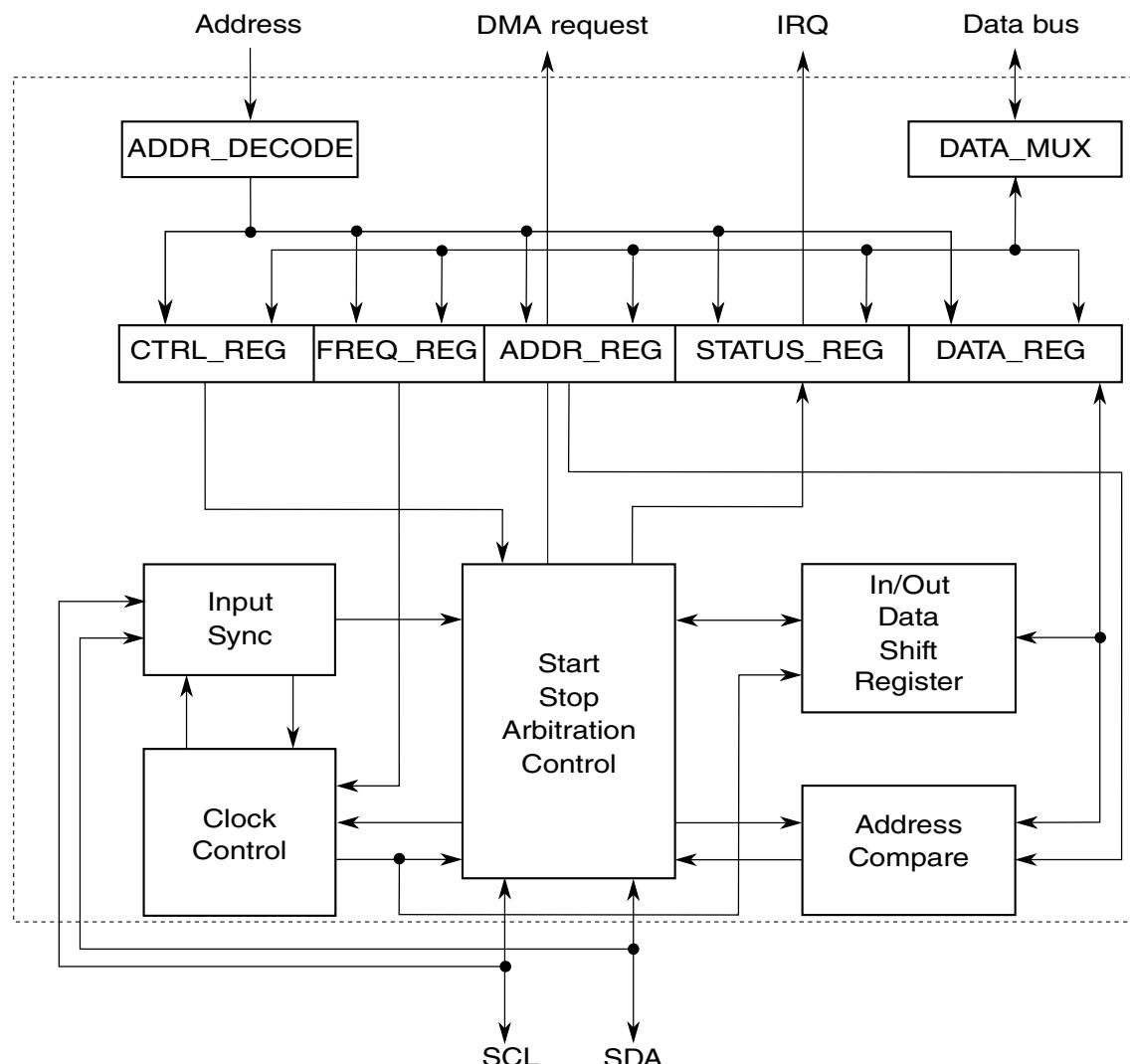
### 26.3.2 Advantages of the I<sup>2</sup>C bus

The synchronous, multiple-master two-wire I<sup>2</sup>C bus:

- Minimizes interconnections between devices
- Allows the connection of additional devices to the bus for expansion and system development
- Includes collision detection and arbitration that prevent data corruption if two or more masters attempt to control the bus simultaneously
- Does not require an external address decoder

### 26.3.3 Module block diagram

The following figure shows a block diagram of the I<sup>2</sup>C module.

Figure 26-1. I<sup>2</sup>C block diagram

### 26.3.4 Features

The I<sup>2</sup>C module has the following key features:

- Compatible with I<sup>2</sup>C bus standard<sup>1</sup>
- Operating speeds
  - Up to 100 kbps in Standard Mode
  - Up to 400 kbps in Fast Mode

1. Compliant with I<sup>2</sup>C 2.0 standard with the exception that HS (high speed) mode is not supported

- Operation at higher baud rates (up to a maximum of module clock/20) with reduced bus loading
- Actual baud rate dependent on the SCL rise time (which depends on external pull-up resistor values and bus loading)
- Multi-master operation
- Software programmable for one of 256 different serial clock frequencies
- Software selectable acknowledge bit
- Interrupt driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection
- Basic DMA interface
- Maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF

### 26.3.5 Modes of operation

The I<sup>2</sup>C module supports the chip modes described in the following table.

**Table 26-4. Chip modes supported by the I<sup>2</sup>C module**

Chip mode	Description	Important notes
RUN	Basic mode of operation	—
DOZE	A low-power mode that allows the system to turn off the clock depending on the state of an internal bit	The I <sup>2</sup> C module can enter this mode when there are no active transfers (active data between valid START and STOP conditions) on the bus.
STOP	The lowest-power mode that allows the chip to turn off all the clocks to the I <sup>2</sup> C module	The I <sup>2</sup> C module can enter this mode when there are no active transfers (active data between valid START and STOP conditions) on the bus. See <a href="#">STOP mode</a> .

In addition to chip modes, the I<sup>2</sup>C module has several module-specific modes. These are described in the following table.

**Table 26-5. Module-specific modes supported by the I<sup>2</sup>C module**

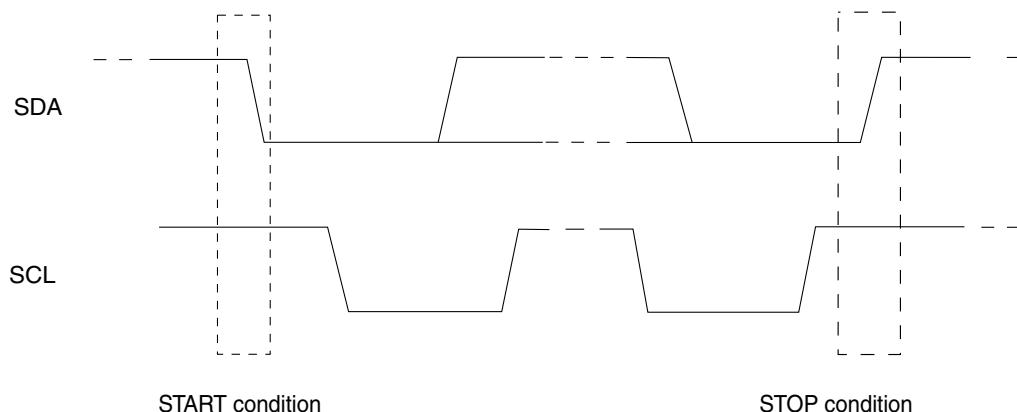
Module mode	Description	Important notes
Master mode	The I <sup>2</sup> C module is the driver of the SDA line.	<ul style="list-style-type: none"> <li>Do not use the I<sup>2</sup>C module's slave address as a calling address.</li> <li>The I<sup>2</sup>C module cannot be a master and a slave simultaneously.</li> </ul>
Slave mode	The I <sup>2</sup> C module is not the driver of the SDA line.	<ul style="list-style-type: none"> <li>Enable the I<sup>2</sup>C module before a START condition from a non-I<sup>2</sup>C master is detected.</li> <li>By default the I<sup>2</sup>C module performs as a slave receiver.</li> </ul>

### 26.3.6 Definition: I<sup>2</sup>C conditions

The following table shows the I<sup>2</sup>C-specific conditions defined for the I<sup>2</sup>C module.

**Table 26-6. I<sup>2</sup>C Conditions**

Condition	Description
START	A condition that denotes the beginning of a new data transfer and awakens all slaves. Each data transfer contains several bytes of data. It is defined as a high-to-low transition of SDA while SCL is high, as shown in the following figure.
STOP	A condition generated by the master to terminate a transfer and free the bus. It is defined as a low-to-high transition of SDA while SCL is high, as shown in the following figure.
Repeated START	A START condition that is generated without a STOP condition to terminate the previous transfer.



**Figure 26-2. START and STOP conditions**

## 26.4 External signal descriptions

This section presents the following topics:

- Signal overview
- Detailed external signal descriptions

### 26.4.1 Signal overview

The I<sup>2</sup>C module uses the Serial Data (SDA) and Serial Clock (SCL) signals as a communication interconnect with other devices. The signal patterns driven on the SDA signal represent address, data, or read/write information at different stages of the protocol.

All devices connected to the SDA and SCL signals must have open-drain or open-collector outputs. The logical AND function is performed on both signals with external pull-up resistors. For the electrical characteristics of these signals, see the data sheet for this chip.

### 26.4.2 Detailed external signal descriptions

The SDA and SCL signals are described in the following table.

**Table 26-7. External signal descriptions**

Signal	Description
SCL	Bidirectional serial clock line of the module, compatible with the I <sup>2</sup> C bus specification
SDA	Bidirectional serial data line of the module, compatible with the I <sup>2</sup> C bus specification

## 26.5 Memory map and register definition

This section provides a detailed description of all memory-mapped registers in the I<sup>2</sup>C module. It presents the following topics:

- Register accessibility
- Register figure conventions

## Memory map and register definition

- I<sup>2</sup>C Bus Address Register (I2C\_IBAD)
- I<sup>2</sup>C Bus Frequency Divider Register (I2C\_IBFD)
- I<sup>2</sup>C Bus Control Register (I2C\_IBCR)
- I<sup>2</sup>C Bus Status Register (I2C\_IBSR)
- I<sup>2</sup>C Bus Data I/O Register (I2C\_IBDR)
- I<sup>2</sup>C Bus Interrupt Config Register (I2C\_IBIC)

### 26.5.1 Register accessibility

Address location 0x0007 is a reserved location, but access to this location will not generate any bus error.

All the I<sup>2</sup>C registers are one byte wide. Reads and writes to these registers must be byte-wide operations.

### 26.5.2 Register figure conventions

The register figures show the field structure using the conventions in the following figure.

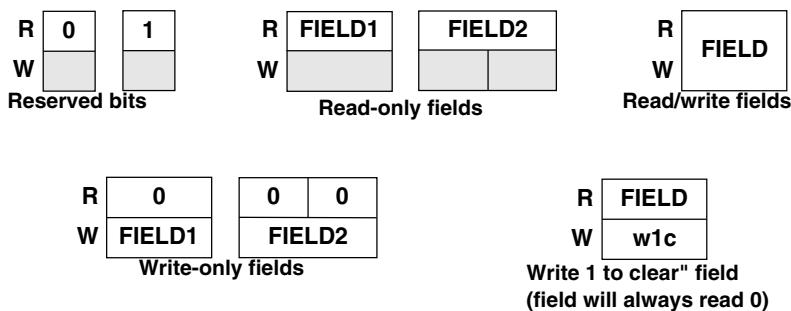


Figure 26-3. Register figure convention

The memory map for the I<sup>2</sup>C module is given below. The total address for each register is the sum of the base address for the I<sup>2</sup>C module and the address offset for each register.

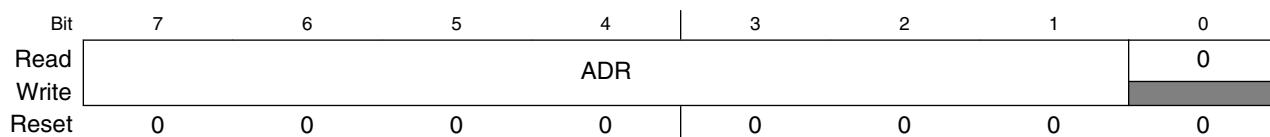
**I<sup>2</sup>C memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_0000	I <sup>2</sup> C Bus Address Register (I2C1_IBAD)	8	R/W	00h	<a href="#">26.5.3/1405</a>
218_0001	I <sup>2</sup> C Bus Frequency Divider Register (I2C1_IBFD)	8	R/W	00h	<a href="#">26.5.4/1406</a>
218_0002	I <sup>2</sup> C Bus Control Register (I2C1_IBCR)	8	R/W	80h	<a href="#">26.5.5/1406</a>
218_0003	I <sup>2</sup> C Bus Status Register (I2C1_IBSR)	8	R/W	80h	<a href="#">26.5.6/1408</a>
218_0004	I <sup>2</sup> C Bus Data I/O Register (I2C1_IBDR)	8	R/W	00h	<a href="#">26.5.7/1409</a>
218_0005	I <sup>2</sup> C Bus Interrupt Config Register (I2C1_IBIC)	8	R/W	00h	<a href="#">26.5.8/1410</a>
219_0000	I <sup>2</sup> C Bus Address Register (I2C2_IBAD)	8	R/W	00h	<a href="#">26.5.3/1405</a>
219_0001	I <sup>2</sup> C Bus Frequency Divider Register (I2C2_IBFD)	8	R/W	00h	<a href="#">26.5.4/1406</a>
219_0002	I <sup>2</sup> C Bus Control Register (I2C2_IBCR)	8	R/W	80h	<a href="#">26.5.5/1406</a>
219_0003	I <sup>2</sup> C Bus Status Register (I2C2_IBSR)	8	R/W	80h	<a href="#">26.5.6/1408</a>
219_0004	I <sup>2</sup> C Bus Data I/O Register (I2C2_IBDR)	8	R/W	00h	<a href="#">26.5.7/1409</a>
219_0005	I <sup>2</sup> C Bus Interrupt Config Register (I2C2_IBIC)	8	R/W	00h	<a href="#">26.5.8/1410</a>
21A_0000	I <sup>2</sup> C Bus Address Register (I2C3_IBAD)	8	R/W	00h	<a href="#">26.5.3/1405</a>
21A_0001	I <sup>2</sup> C Bus Frequency Divider Register (I2C3_IBFD)	8	R/W	00h	<a href="#">26.5.4/1406</a>
21A_0002	I <sup>2</sup> C Bus Control Register (I2C3_IBCR)	8	R/W	80h	<a href="#">26.5.5/1406</a>
21A_0003	I <sup>2</sup> C Bus Status Register (I2C3_IBSR)	8	R/W	80h	<a href="#">26.5.6/1408</a>
21A_0004	I <sup>2</sup> C Bus Data I/O Register (I2C3_IBDR)	8	R/W	00h	<a href="#">26.5.7/1409</a>
21A_0005	I <sup>2</sup> C Bus Interrupt Config Register (I2C3_IBIC)	8	R/W	00h	<a href="#">26.5.8/1410</a>
21B_0000	I <sup>2</sup> C Bus Address Register (I2C4_IBAD)	8	R/W	00h	<a href="#">26.5.3/1405</a>
21B_0001	I <sup>2</sup> C Bus Frequency Divider Register (I2C4_IBFD)	8	R/W	00h	<a href="#">26.5.4/1406</a>
21B_0002	I <sup>2</sup> C Bus Control Register (I2C4_IBCR)	8	R/W	80h	<a href="#">26.5.5/1406</a>
21B_0003	I <sup>2</sup> C Bus Status Register (I2C4_IBSR)	8	R/W	80h	<a href="#">26.5.6/1408</a>
21B_0004	I <sup>2</sup> C Bus Data I/O Register (I2C4_IBDR)	8	R/W	00h	<a href="#">26.5.7/1409</a>
21B_0005	I <sup>2</sup> C Bus Interrupt Config Register (I2C4_IBIC)	8	R/W	00h	<a href="#">26.5.8/1410</a>

**26.5.3 I<sup>2</sup>C Bus Address Register (I2Cx\_IBAD)**

This register contains the address the I<sup>2</sup>C Bus will respond to when addressed as a slave. This is not the address sent on the bus during the address transfer.

Address: Base address + 0h offset



## Memory map and register definition

### I2Cx\_IBAD field descriptions

Field	Description
7–1 ADR	Slave Address. Specific slave address to be used by the I <sup>2</sup> C Bus module. <b>NOTE:</b> The default mode of I <sup>2</sup> C Bus is slave mode for an address match on the bus.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

## 26.5.4 I2C Bus Frequency Divider Register (I2Cx\_IBFD)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read	IBC							
Write	0	0	0	0	0	0	0	0

### I2Cx\_IBFD field descriptions

Field	Description
IBC	I-Bus Clock Rate. This field is used to prescale the bus clock for bit rate selection. See <a href="#">Clock rate and IBFD settings</a> .

## 26.5.5 I2C Bus Control Register (I2Cx\_IBCR)

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read	MDIS				IBIE		MSSL	
Write	1	0	0	0	NOACK	0	DMAEN	IBDOZE

### I2Cx\_IBCR field descriptions

Field	Description
7 MDIS	Module disable. This bit controls the software reset of the entire I <sup>2</sup> C Bus module. <b>NOTE:</b> If the I <sup>2</sup> C Bus module is enabled in the middle of a byte transfer, the interface behaves as follows: slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected. Master mode will not be aware that the bus is busy, hence if a start cycle is initiated then the current bus cycle may become corrupt. This would ultimately result in either the current bus master or the I <sup>2</sup> C Bus module losing arbitration, after which, bus operation would return to normal. 0 The I <sup>2</sup> C Bus module is enabled. This bit must be cleared before any other IBCR bits have any effect 1 The module is reset and disabled. This is the power-on reset situation. When high, the interface is held in reset, but registers can still be accessed. Status register bits (IBSR) are not valid when module is disabled.

Table continues on the next page...

**I<sup>2</sup>Cx\_IBCR field descriptions (continued)**

Field	Description
6 IBIE	<p>I-Bus Interrupt Enable.</p> <p>0 Interrupts from the I<sup>2</sup>C Bus module are disabled. This does not clear any currently pending interrupt condition.</p> <p>1 Interrupts from the I<sup>2</sup>C Bus module are enabled. An I<sup>2</sup>C Bus interrupt occurs provided the IBIF bit in the status register is also set.</p>
5 MSSL	<p>Master/Slave mode select. When this bit is changed from 0 to 1, a START signal is generated on the bus and the master mode is selected. When this bit is changed from 1 to 0, a STOP signal is generated and the operation mode changes from master to slave. A STOP signal should be generated only if the IBIF flag is set. This field is cleared without generating a STOP signal when the master loses arbitration.</p> <p>0 Slave Mode</p> <p>1 Master Mode</p>
4 TXRX	<p>Transmit/Receive mode select. This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the status register. In master mode this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit will always be high.</p> <p>0 Receive</p> <p>1 Transmit</p>
3 NOACK	<p>Data Acknowledge disable. This bit specifies the value driven onto SDA during data acknowledge cycles for both master and slave receivers. The I<sup>2</sup>C module will always acknowledge address matches, provided it is enabled, regardless of the value of NOACK.</p> <p><b>NOTE:</b> Values written to this bit are only used when the I<sup>2</sup>C Bus is a receiver, not a transmitter.</p> <p>0 An acknowledge signal will be sent out to the bus at the 9th clock bit after receiving one byte of data</p> <p>1 No acknowledge signal response is sent (i.e., acknowledge bit = 1)</p>
2 RSTA	<p>Repeat Start. Writing a one to this bit will generate a repeated START condition on the bus, provided it is the current bus master. This bit will always be read as a low. Attempting a repeated start at the wrong time, if the bus is owned by another master, will result in loss of arbitration.</p> <p>0 No effect</p> <p>1 Generate repeat start cycle</p>
1 DMAEN	<p>DMA Enable. When this bit is set, the DMA Tx and Rx lines will be asserted when the I<sup>2</sup>C module requires data to be read or written to the data register. No Transfer Done interrupts will be generated when this bit is set, however an interrupt will be generated if the loss of arbitration or addressed as slave conditions occur. The DMA mode is only valid when the I<sup>2</sup>C module is configured as a Master and the DMA transfer still requires CPU intervention at the start and the end of each frame of data. See the DMA Application Information section for more details.</p> <p>0 Disable the DMA TX/RX request signals</p> <p>1 Enable the DMA TX/RX request signals</p>
0 IBDOZE	<p>I-Bus Interface Stop in DOZE mode.</p> <p>If the IBDOZE mode is SET, the I<sup>2</sup>C module will enter DOZE mode when the DOZE signal is asserted, if there are no current transactions on the bus. The I<sup>2</sup>C module would then signal to the system that the clock can be shut down.</p> <p>If the IBDOZE bit is cleared when the DOZE signal is asserted, the I<sup>2</sup>C Bus module clock remains alive, and any current transactions continue as normal.</p>

*Table continues on the next page...*

**I2Cx\_IBCR field descriptions (continued)**

Field	Description
	0 I <sup>2</sup> C Bus module clock operates normally 1 Halt I <sup>2</sup> C Bus module clock generation (if DOZE mode signal asserted)

**26.5.6 I2C Bus Status Register (I2Cx\_IBSR)**

Address: Base address + 3h offset

Bit	7	6	5	4	3	2	1	0
Read	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
Write				w1c			w1c	
Reset	1	0	0	0	0	0	0	0

**I2Cx\_IBSR field descriptions**

Field	Description
7 TCF	<p>Transfer complete.</p> <p>While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the 9th clock of a byte transfer.</p> <p><b>NOTE:</b> This bit is only valid during or immediately following a transfer to the I<sup>2</sup>C module or from the I<sup>2</sup>C module.</p> <p>0 Transfer in progress 1 Transfer complete</p>
6 IAAS	<p>Addressed as a slave.</p> <p>When its own specific address (I-Bus Address Register) is matched with the calling address, this bit is set. The CPU is interrupted provided the IBIE is set. Then the CPU needs to check the SRW bit and set the TXRX field accordingly. Writing to the I-Bus Control Register clears this bit.</p> <p>0 Not addressed 1 Addressed as a slave</p>
5 IBB	<p>Bus busy.</p> <p>This bit indicates the status of the bus. When a START signal is detected, IBB is set. If a STOP signal is detected, IBB is cleared and the bus enters idle state.</p> <p><b>NOTE:</b> Software must ensure that the I<sup>2</sup>C bus is idle by checking the IBSR[IBB] field (bus busy) before switching to master mode and attempting a START cycle.</p> <p>0 Bus is Idle 1 Bus is busy</p>
4 IBAL	<p>Arbitration Lost.</p> <p>The arbitration lost bit (IBAL) is set by hardware when the arbitration procedure is lost. Arbitration is lost in the following circumstances:</p> <ul style="list-style-type: none"> <li>• SDA is sampled low when the master drives a high during an address or data transmit cycle.</li> <li>• SDA is sampled low when the master drives a high during the acknowledge bit of a data receive cycle.</li> <li>• A start cycle is attempted when the bus is busy.</li> </ul>

Table continues on the next page...

**I2Cx\_IBSR field descriptions (continued)**

Field	Description				
	<ul style="list-style-type: none"> <li>A repeated start cycle is requested in slave mode.</li> <li>A stop condition is detected when the master did not request it.</li> </ul> <p>This bit must be cleared by software, by writing a one to it. A write of zero has no effect.</p>				
3 Reserved	<p>This field is reserved. This read-only field is reserved and always has the value 0.</p>				
2 SRW	<p>Slave Read/Write. When the IAAS bit is set, this bit indicates the value of the R/W command bit of the calling address sent from the master. This bit is only valid when the I-Bus is in slave mode, a complete address transfer has occurred with an address match and no other transfers have been initiated. By reading this field, the CPU can detect slave transmit/receive mode according to the command of the master.</p> <table> <tr> <td>0</td> <td>Slave receive, master writing to slave</td> </tr> <tr> <td>1</td> <td>Slave transmit, master reading from slave</td> </tr> </table>	0	Slave receive, master writing to slave	1	Slave transmit, master reading from slave
0	Slave receive, master writing to slave				
1	Slave transmit, master reading from slave				
1 IBIF	<p>I-Bus Interrupt Flag. The IBIF bit is set when one of the following conditions occurs:</p> <ul style="list-style-type: none"> <li>Arbitration lost (IBAL bit set)</li> <li>Byte transfer complete (TCF bit set and DMAEN bit not set)</li> <li>Addressed as slave (IAAS bit set)</li> <li>NoAck from Slave (MS &amp; Tx bits set)</li> <li>I<sup>2</sup>C Bus going idle (IBB high-low transition and enabled by BIIE)</li> </ul> <p>A processor interrupt request will be caused if the IBIE bit is set. This bit must be cleared by software, by writing a one to it. A write of zero has no effect on this bit. In DMA mode (DMAEN set) a byte transfer complete condition will not trigger the setting of IBIF. All other conditions still apply.</p>				
0 RXAK	<p>Received Acknowledge. This is the value of SDA during the acknowledge bit of a bus cycle. If the received acknowledge bit (RXAK) is low, it indicates an acknowledge signal has been received after the completion of 8 bits data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the 9th clock. This bit is valid only after transfer is complete.</p> <table> <tr> <td>0</td> <td>Acknowledge received</td> </tr> <tr> <td>1</td> <td>No acknowledge received</td> </tr> </table>	0	Acknowledge received	1	No acknowledge received
0	Acknowledge received				
1	No acknowledge received				

**26.5.7 I2C Bus Data I/O Register (I2Cx\_IBDR)**

In master transmit mode, when data is written to the IBDR, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates next byte data receiving. In slave mode, the same functions are available after an address match has occurred.

**NOTE**

The IBCR[TXRX] field must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the I<sup>2</sup>C is configured for master transmit but a master receive is desired, then reading the IBDR will not initiate the receive.

## Memory map and register definition

Reading the IBDR will return the most recent byte received while the I<sup>2</sup>C is configured in either master receive or slave receive modes. The IBDR does not reflect every byte that is transmitted on the I<sup>2</sup>C bus, nor can software verify that a byte has been written to the IBDR correctly by reading it back.

In master transmit mode, the first byte of data written to IBDR following assertion of MSSL is used for the address transfer and should comprise the calling address (in position DATA[7:1]) concatenated with the required R/ W bit (in position D0).

### NOTE

When the I<sup>2</sup>C is configured in master mode and receiving data from a slave that is transmitting data bytes on an irregular basis, the master cannot know whether the data received in the IBDR is the old latched data or the new data received from the slave. To avoid this, 2 consecutive intermittent data bytes from slave should be different.

Address: Base address + 4h offset

Bit	7	6	5	4		3	2	1	0
Read						DATA			
Write	0	0	0	0		0	0	0	0

### I2Cx\_IBDR field descriptions

Field	Description								
DATA	Data transmitted or received								

## 26.5.8 I2C Bus Interrupt Config Register (I2Cx\_IBIC)

To program BIIE = 1, you must ensure that IBCR[MDIS] = 0.

Address: Base address + 5h offset

Bit	7	6	5	4		3	2	1	0
Read	BIIE	BYTERXIE				0			
Write	0	0	0	0		0	0	0	0

### I2Cx\_IBIC field descriptions

Field	Description								
7 BIIE	Bus Idle Interrupt Enable bit. This config bit can be used to enable the generation of an interrupt once the I <sup>2</sup> C bus becomes idle. Once this bit is set, an IBB high-low transition will set the IBIF bit. This feature can be used to signal to the CPU the completion of a STOP on the I <sup>2</sup> C bus.								

*Table continues on the next page...*

**I<sup>2</sup>Cx\_IBIC field descriptions (continued)**

Field	Description
	0 Bus Idle Interrupts disabled 1 Bus Idle Interrupts enabled
6 BYTERXIE	Byte receive interrupt enable  This field is used to generate an interrupt every time the I <sup>2</sup> C master/slave receives a new byte. This feature can be useful when an I <sup>2</sup> C master is receiving data from a slave that is transmitting on an irregular basis.  BYTERXIE is updated only when the I <sup>2</sup> C is enabled (IBCR[MDIS]=0).
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

## 26.6 Functional description

This section presents the following topics:

- [Notes about module operation](#)
- [Transactions](#)
- [Arbitration procedure](#)
- [Clock behavior](#)
- [Interrupts](#)
- [DMA interface](#)

### 26.6.1 Notes about module operation

- The I<sup>2</sup>C module always performs as a slave receiver by default, unless explicitly programmed to be a master or slave transmitter.
- When the I<sup>2</sup>C module is acting as a master, it must not try to call its own slave address.

### 26.6.2 Transactions

This section presents the following topics:

- [Protocol overview](#)

## Functional description

- Transaction protocol definitions
- High-level protocol steps
- START condition
- Slave address transmission
- Data transmission
- STOP condition
- Repeated START condition

### 26.6.2.1 Protocol overview

The following figure shows the behavior of SCL and SDA during a typical I<sup>2</sup>C transaction.

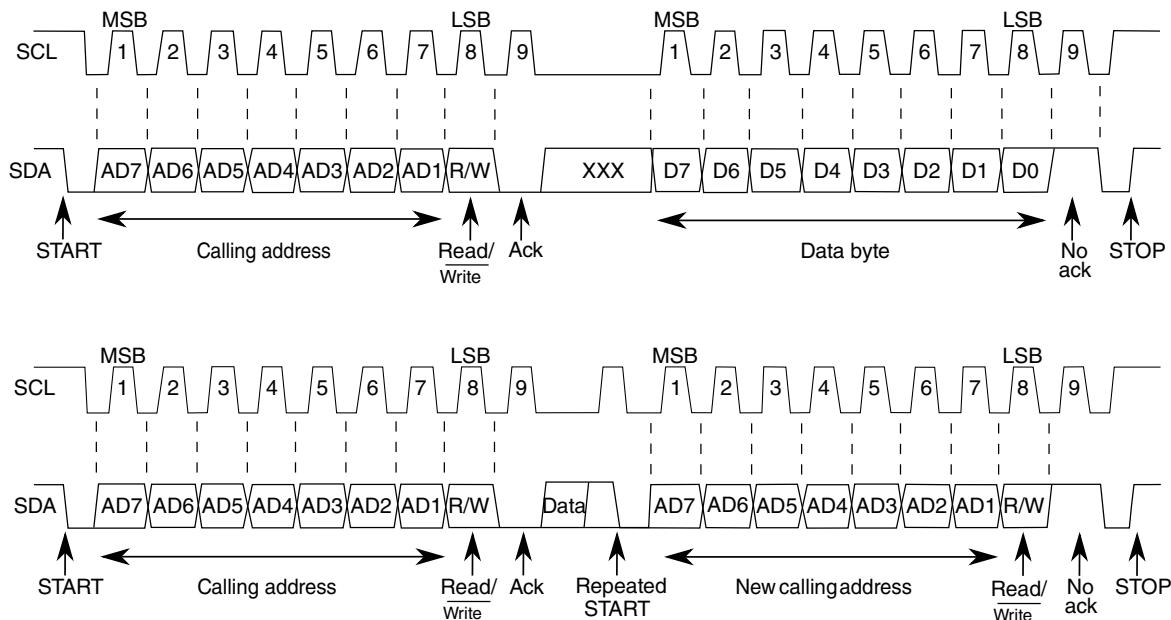


Figure 26-4. I<sup>2</sup>C transaction protocol

### 26.6.2.2 Transaction protocol definitions

This section defines several important terms presented in Figure 26-4.

**Table 26-8. I<sup>2</sup>C definitions**

Term	Definition
START	A START condition, as defined in <a href="#">Section 1.2.6, Definition: I<sup>2</sup>C conditions</a> "
STOP	A STOP condition, as defined in <a href="#">Section 1.2.6, Definition: I<sup>2</sup>C conditions</a> "
Calling (slave) address	A seven-bit address used to identify a slave on the I <sup>2</sup> C bus. The requirements for specifying this address are presented in <a href="#">Section 1.5.2.3, I<sup>2</sup>C calling address requirements</a> ."
Read/write (R/W)	A bit that specifies the direction of the data transfer to the slave as follows: <ul style="list-style-type: none"> <li>• 0=The data is being transferred from the master to the slave ("write")</li> <li>• 1=The data is being transferred from the slave to the master ("read")</li> </ul>
Ack	A bit that specifies the acknowledgement of a calling address, indicated by pulling SDA low.

### 26.6.2.3 I<sup>2</sup>C calling address requirements

The calling addresses of the devices used on an I<sup>2</sup>C network are subject to the following requirements:

- Each slave must have a unique calling address.
- A master must not transmit a calling address that is the same as its own slave address.

### 26.6.2.4 High-level protocol steps

The I<sup>2</sup>C protocol conceptually supports two types of transfers, which are illustrated in [Figure 26-4](#). The significant steps in these transfers are presented in the following table. Details of each of these steps are presented in subsequent sections.

**Table 26-9. I<sup>2</sup>C high-level protocol steps**

Standard Transfer	Repeated START Transfer
1. START condition	1. START condition
2. Slave target or general call address transmission	2. Slave target or general call address transmission
3. Acknowledgment from slave	3. Acknowledgment from slave
4. Data transfer	4. Data transfer
5. STOP condition	5. Repeated START condition
6. (repeat Steps 1–4)	6. (repeat Steps 2–4 as needed)
	7. STOP condition.
	8. (repeat Steps 1–7)

### 26.6.2.5 START condition

When the bus is free, that is, no master device is engaging the bus (both SCL and SDA lines are at logical high), a master may initiate communication by sending a START condition (see [Definition: I<sup>2</sup>C conditions](#)). This signal denotes the beginning of a new data transfer (each data transfer may contain several bytes of data) and brings all slaves out of their idle states.

See [Clock rate and IBFD settings](#) and [I<sup>2</sup>C Bus Frequency Divider Register \(I<sup>2</sup>C\\_IBFD\)](#) for the associated timing requirements.

### 26.6.2.6 Slave address transmission

The master transmits the slave address on the next clock cycle after the START condition (see [START condition](#)). The process of slave address transmission is presented in the following table.

**Table 26-10. Slave address transmission process**

Step	Action
1	The master transmits the seven-bit slave address.
2	The master transmits the R/W bit.
3	Each slave examines the transmitted address and compares it to its own. If the addresses match, the slave device returns the acknowledge bit on the ninth SCL clock cycle.
4	The master waits for the acknowledge bit and determines the next step as follows: <ul style="list-style-type: none"> <li>The acknowledge bit is set: The master must initiate a data transfer followed by either a STOP condition or a repeated START condition.</li> <li>The acknowledge bit is cleared: The master must wait for SCL to return to logic zero.</li> </ul>

### 26.6.2.7 Data transmission

A data transfer session has the following characteristics:

- Can transmit one or more bytes of data
- Awakens all slaves
- Proceeds on a byte-by-byte basis in the direction specified by the R/W bit sent by the calling master

The transmitted data is subject to the following requirements:

- Each data byte must consist of 8 bits.
- Data bits can be changed only while SCL is low and must be held stable while SCL is high.
- One data bit is transmitted during one SCL clock pulse.
- The most significant bit (msb) must be transmitted first.
- Each data byte must be followed by an acknowledge bit on the ninth SCL clock pulse.

If the slave receiver does not acknowledge the master, the SDA line must be left high by the slave. The master can then generate a stop condition to abort the data transfer or a START condition (repeated START) to begin a new calling.

If the master receiver does not acknowledge the slave transmitter after a byte of transmission, the slave interprets that the end-of-data has been reached. Then the slave releases the SDA line for the master to generate a STOP or a START condition.

### 26.6.2.8 STOP condition

The master can terminate the communication by generating a STOP condition (see [Definition: I<sup>2</sup>C conditions](#)). It can do so even if the slave has generated an acknowledge, at which point the slave must release the bus.

A master is not required to send a STOP condition at the end of every transfer. For more information, see [Repeated START condition](#).

See [Clock rate and IBFD settings](#) and [I<sup>2</sup>C Bus Frequency Divider Register \(I<sup>2</sup>C\\_IBFD\)](#) for the associated timing requirements.

### 26.6.2.9 Repeated START condition

The I<sup>2</sup>C protocol also supports a repeated START condition, which can be generated without a preceding STOP condition. A master device can use this condition to communicate with another slave or with the same slave in a different mode without releasing the bus. This condition is illustrated in the second timing diagram of [Figure 26-4](#).

### 26.6.3 Arbitration procedure

The I<sup>2</sup>C bus is a true multi-master bus that allows more than one master to be connected to it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic "1" while another master transmits logic "0". The losing masters immediately switch over to slave mode and stop driving the SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, a status bit is set by hardware to indicate loss of arbitration.

### 26.6.4 Clock behavior

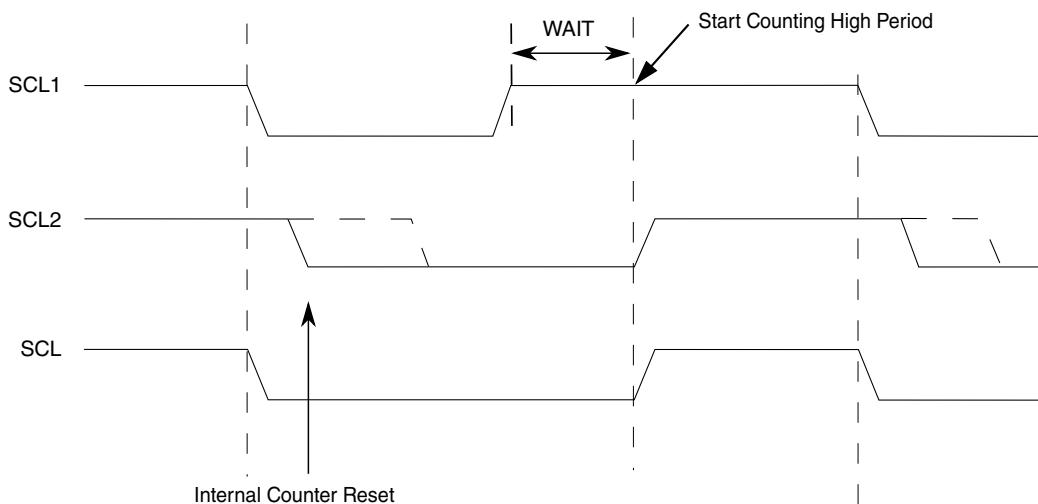
This section presents the following topics:

- [Clock synchronization](#)
- [Clock stretching](#)
- [Handshaking](#)
- [Clock rate and IBFD settings](#)

#### 26.6.4.1 Clock synchronization

Due to the wired-AND logic on the SCL line, a high-to-low transition on the SCL line affects all devices connected on the bus. The devices begin counting their low period when the master drives the SCL line low. After a device has driven SCL low, it holds the SCL line low until the clock high state is reached.

However, the change of low-to-high in a device clock may not change the state of the SCL line if another device is still within its low period. Therefore, the synchronized clock signal, SCL, is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see the following figure). When all devices concerned have counted off their low period, the synchronized SCL line is released and pulled high. Then there is no difference between the devices' clocks and the state of the SCL line, and all the devices begin counting their high periods. The first device to complete its high period pulls the SCL line low again.

**Figure 26-5. I<sup>2</sup>C bus clock synchronization**

### 26.6.4.2 Clock stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master has driven SCL low, the slave can drive SCL low for the required period and then release it. If the slave SCL low period is greater than the master SCL low period then the resulting SCL bus signal low period is stretched.

### 26.6.4.3 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (9 bits). In such cases, it halts the bus clock and forces the master clock into wait state until the slave releases the SCL line.

### 26.6.4.4 Clock rate and IBFD settings

#### 26.6.4.4.1 Timing definitions

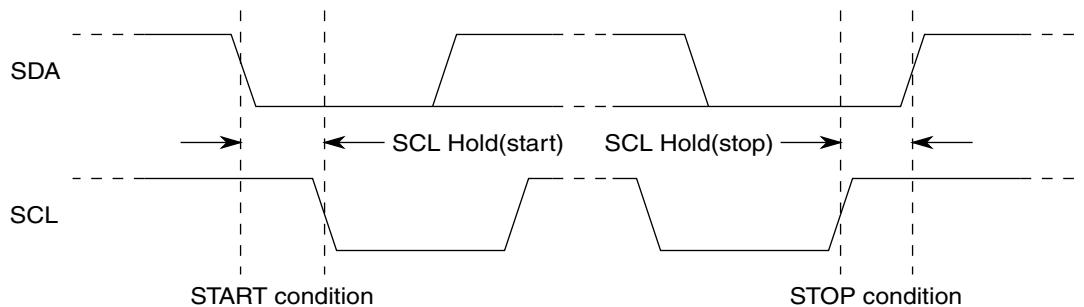
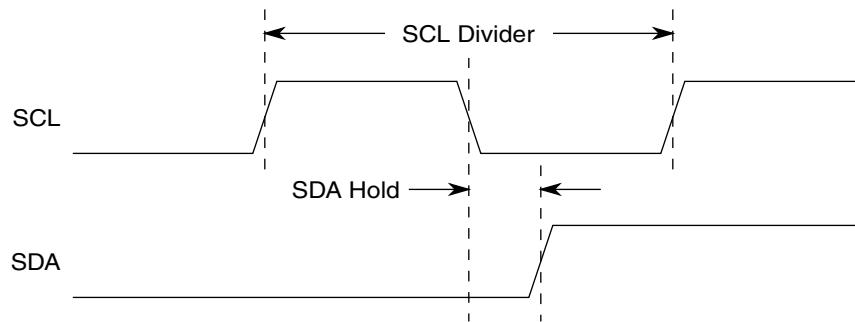
**Table 26-11. Timing definitions relevant to clock rate and IBFD settings**

Term	Definition
SCL Divider	The factor used to prescale the CPU clock for bit rate selection. This is relevant to: <ul style="list-style-type: none"> <li>• <a href="#">Figure 26-6</a></li> <li>• <a href="#">Table 26-13</a></li> </ul>

*Table continues on the next page...*

**Table 26-11. Timing definitions relevant to clock rate and IBFD settings (continued)**

Term	Definition
SCL period	(CPU clock period) × (SCL Divider)
SCL Hold	The factor used to prescale the CPU clock for bit rate selection. This is relevant to: <ul style="list-style-type: none"> <li>• <a href="#">Figure 26-6</a></li> <li>• <a href="#">Table 26-13</a></li> </ul>
SDA Hold	The factor used to prescale the CPU clock for bit rate selection. This is relevant to: <ul style="list-style-type: none"> <li>• <a href="#">Figure 26-7</a></li> <li>• <a href="#">Table 26-13</a></li> </ul>

**Figure 26-6. SCL Divider and SDA Hold****Figure 26-7. SDA Hold time**

#### 26.6.4.4.2 Divider and hold values

You have up to three MUL options available for all divider values, as shown in [Table 26-12](#). Your choice of MUL determines the internal monitor rate of the I<sup>2</sup>C bus (SCL and SDA signals):

- A lower MUL value results in a higher sampling rate of the I<sup>2</sup>C signals.
- A higher MUL value results in a lower sampling rate of the I<sup>2</sup>C signals. This gives the I<sup>2</sup>C module greater immunity against glitches in the I<sup>2</sup>C signals.

**Table 26-12. MUL values as a function of IBC**

When IBC is...	MUL is...
00h–3Fh	1
40h–7Fh	2
80h–BFh	4

**Table 26-13. I<sup>2</sup>C divider and hold values**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
00	20	7	6	11
01	22	7	7	12
02	24	8	8	13
03	26	8	9	14
04	28	9	10	15
05	30	9	11	16
06	34	10	13	18
07	40	10	16	21
08	28	7	10	15
09	32	7	12	17
0A	36	9	14	19
0B	40	9	16	21
0C	44	11	18	23
0D	48	11	20	25
0E	56	13	24	29
0F	68	13	30	35
10	48	9	18	25
11	56	9	22	29
12	64	13	26	33
13	72	13	30	37
14	80	17	34	41
15	88	17	38	45
16	104	21	46	53
17	128	21	58	65
18	80	9	38	41
19	96	9	46	49
1A	112	17	54	57
1B	128	17	62	65
1C	144	25	70	73
1D	160	25	78	81
1E	192	33	94	97

*Table continues on the next page...*

**Table 26-13. I<sup>2</sup>C divider and hold values (continued)**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
1F	240	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
40	40	14	12	22
41	44	14	14	24
42	48	16	16	26
43	52	16	18	28
44	56	18	20	30

*Table continues on the next page...*

**Table 26-13. I<sup>2</sup>C divider and hold values (continued)**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
45	60	18	22	32
46	68	20	26	36
47	80	20	32	42
48	56	14	20	30
49	64	14	24	34
4A	72	18	28	38
4B	80	18	32	42
4C	88	22	36	46
4D	96	22	40	50
4E	112	26	48	58
4F	136	26	60	70
50	96	18	36	50
51	112	18	44	58
52	128	26	52	66
53	144	26	60	74
54	160	34	68	82
55	176	34	76	90
56	208	42	92	106
57	256	42	116	130
58	160	18	76	82
59	192	18	92	98
5A	224	34	108	114
5B	256	34	124	130
5C	288	50	140	146
5D	320	50	156	162
5E	384	66	188	194
5F	480	66	236	242
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258
64	576	98	284	290
65	640	98	316	322
66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450

*Table continues on the next page...*

**Table 26-13. I<sup>2</sup>C divider and hold values (continued)**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074
7F	7680	1026	3836	3842
80	80	28	24	44
81	88	28	28	48
82	96	32	32	52
83	104	32	36	56
84	112	36	40	60
85	120	36	44	64
86	136	40	52	72
87	160	40	64	84
88	112	28	40	60
89	128	28	48	68
8A	144	36	56	76
8B	160	36	64	84
8C	176	44	72	92
8D	192	44	80	100
8E	224	52	96	116
8F	272	52	120	140
90	192	36	72	100

Table continues on the next page...

**Table 26-13. I<sup>2</sup>C divider and hold values (continued)**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
91	224	36	88	116
92	256	52	104	132
93	288	52	120	148
94	320	68	136	164
95	352	68	152	180
96	416	84	184	212
97	512	84	232	260
98	320	36	152	164
99	384	36	184	196
9A	448	68	216	228
9B	512	68	248	260
9C	576	100	280	292
9D	640	100	312	324
9E	768	132	376	388
9F	960	132	472	484
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964
A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284
B1	3072	260	1528	1540
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076

*Table continues on the next page...*

**Table 26-13. I<sup>2</sup>C divider and hold values (continued)**

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

## 26.6.5 Interrupts

This section presents the following topics:

- [Interrupt vector](#)
- [Interrupt description](#)

### 26.6.5.1 Interrupt vector

The I<sup>2</sup>C module uses only one interrupt vector.

**Table 26-14. Interrupt summary**

Interrupt	Offset	Vector	Priority	Source	Description
I <sup>2</sup> C Interrupt	—	—	—	IBAL, TCF, IAAS, IBB bits in IBSSR register	When any of IBAL, TCF or IAAS bits is set, an interrupt may be caused based on Arbitration lost, Transfer Complete or Address Detect conditions. If enabled by BIIE, the de-assertion of IBB can also cause an interrupt, indicating that the bus is idle.

### 26.6.5.2 Interrupt description

There are five types of internal interrupts in the I<sup>2</sup>C. The interrupt service routine can determine the interrupt type by reading the Status register.

I<sup>2</sup>C Interrupt can be generated on the following events:

- Arbitration Lost condition (IBAL bit set)
- Byte Transfer condition (TCF bit set and DMAEN bit not set)
- Address Detect condition (IAAS bit set)
- No Acknowledge from slave received when expected
- Bus Going Idle (IBB bit not set)

The I<sup>2</sup>C interrupt is enabled by the IBCR[IBIE] bit. It must be cleared by writing '1' to the IBIF bit in the interrupt service routine. The Bus Going Idle interrupt needs to be additionally enabled by the IBIC[BIIIE] bit.

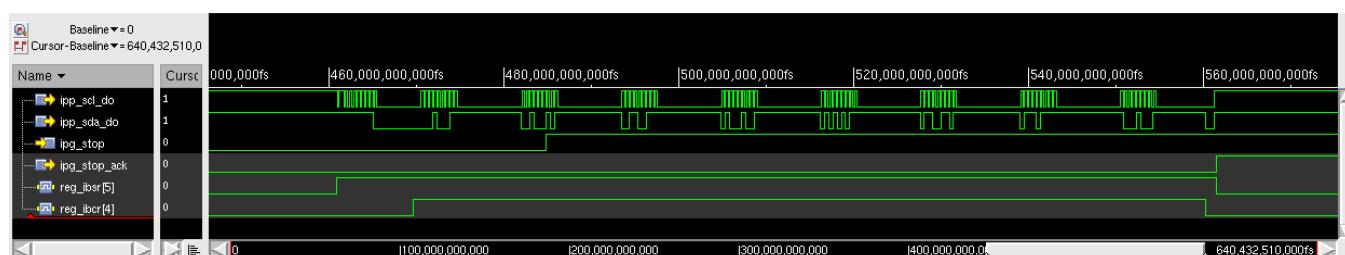
## 26.6.6 STOP mode

This mode allows the software to put the I<sup>2</sup>C module in power-down state. Once the STOP request is asserted, the I<sup>2</sup>C module comes to a graceful halt after completing all the ongoing transactions.

As soon as the I<sup>2</sup>C module enters STOP mode:

- The I<sup>2</sup>C clock is disabled.
- No transaction can take place.
- All registers are inaccessible.

The I<sup>2</sup>C module enters this mode only after successfully completing the current ongoing transaction, hence the low-power request is acknowledged once the STOP condition occurs. The user must ensure that the bus is free when STOP is requested. To request STOP for ongoing transmission the user must wait until the transmission is complete, followed by clearing of the IBCR[TXRX] field. See the figure below for more details.



**Figure 26-8. I<sup>2</sup>C stop mode behavior when master is receiving and slave is transmitting**

## 26.6.7 DMA interface

A simple DMA interface is implemented so that the I<sup>2</sup>C can request data transfers with minimal support from the CPU (see [DMA application information](#)). DMA mode is enabled by setting bit 1 in the Control Register (IBCR).

The DMA interface is operational when the I<sup>2</sup>C module is configured for Master mode.

At least three bytes of data per frame must be transferred from/to the slave when using DMA mode, although in practice it will only be worthwhile using the DMA mode when there is a large number of data bytes to transfer per frame.

Two internal signals, TX request and RX request, are used to signal the DMA controller when the I<sup>2</sup>C module requires data to be written or read from the data register.

Further details of the DMA interface can be found in the [Initialization/application information](#), of this document.

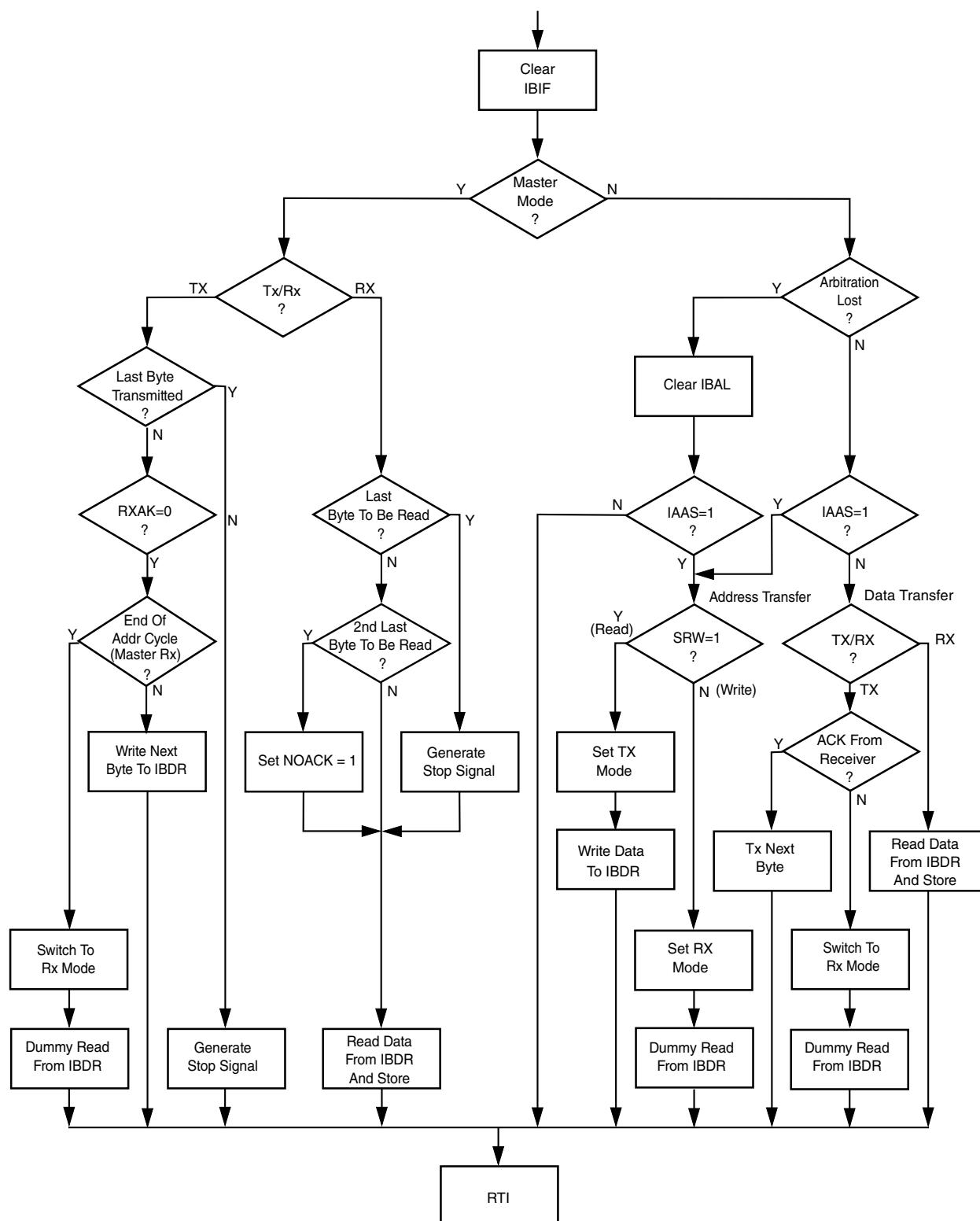
## 26.7 Initialization/application information

This section presents the following topics:

- [Recommended interrupt service flow](#)
- [General programming guidelines \(for both master and slave mode\)](#)
- [Programming guidelines specific to master mode](#)
- [Programming guidelines specific to slave mode](#)
- [DMA application information](#)

### 26.7.1 Recommended interrupt service flow

The following figure shows a flowchart for the recommended I<sup>2</sup>C interrupt service routine. Deviation from the flowchart may result in unpredictable I<sup>2</sup>C bus behavior.

Figure 26-9. Recommended I<sup>2</sup>C interrupt service routine flowchart

## 26.7.2 General programming guidelines (for both master and slave mode)

This section provides programming guidelines recommended for the I<sup>2</sup>C module in both master and slave mode. It presents the following topics:

- Initializing the I<sup>2</sup>C module
- Software response after a transfer

### 26.7.2.1 Initializing the I<sup>2</sup>C module

The following table describes how to initialize the I<sup>2</sup>C module.

**Table 26-15. I<sup>2</sup>C initialization procedure**

Step	Action
1	Use <a href="#">I<sup>2</sup>C Bus Frequency Divider Register (I2C_IBFD)</a> to select the required division ratio to obtain SCL frequency from the system clock.
2	Use <a href="#">I<sup>2</sup>C Bus Address Register (I2C_IBAD)</a> to define the slave address.
3	Clear the MDIS field in <a href="#">I<sup>2</sup>C Bus Control Register (I2C_IBCR)</a> to enable the I <sup>2</sup> C interface system.
4	Use <a href="#">I<sup>2</sup>C Bus Control Register (I2C_IBCR)</a> to select Master/Slave mode, Transmit/Receive mode, and whether interrupts are enabled or disabled.
5	(Optional) Use <a href="#">I<sup>2</sup>C Bus Interrupt Config Register (I2C_IBIC)</a> to further refine the interrupt behavior.

### 26.7.2.2 Software response after a transfer

Transmission or reception of a byte will set the data transferring bit (TCF) to 1, which indicates one byte communication is finished. The I<sup>2</sup>C Bus interrupt bit (IBIF) is set also; an interrupt will be generated if the interrupt function is enabled during initialization by setting the IBIE bit. The IBIF (interrupt flag) can be cleared by writing one (in the interrupt service routine, if interrupts are used).

The TCF bit will be cleared to indicate data transfer in progress whenever data register is written to in transmit mode, or during reading out from data register in receive mode. The TCF bit should not be used as a data transfer complete flag as the flag timing is dependent on a number of factors including the I<sup>2</sup>C bus frequency. This bit may not conclusively provide an indication of a transfer complete situation. It is recommended that transfer complete situations are detected using the IBIF flag.

Software may service the I<sup>2</sup>C I/O in the main program by monitoring the IBIF bit if the interrupt function is disabled. Polling should monitor the IBIF bit rather than the TCF bit since their operation is different when arbitration is lost.

When a "Transfer Complete" interrupt occurs at the end of the address cycle, the master will always be in transmit mode, i.e. the address is transmitted. If master receive mode is required, indicated by R/W bit sent with slave calling address, then the Tx/Rx bit at Master side should be toggled at this stage. If Master does not receive an ACK from Slave, then transmission must be re-initiated or terminated.

In slave mode, IAAS bit will get set in IBSR if Slave address (IBAD) matches the Master calling address. This is an indication that Master-Slave data communication can now start. During address cycles (IBSR[IAAS]=1), the SRW bit in the status register is read to determine the direction of the subsequent transfer and the Tx/Rx bit is programmed accordingly. For slave mode data cycles (IBSR[IAAS]=0), the SRW bit is not valid. The Tx/Rx bit in the control register should be read to determine the direction of the current transfer.

### 26.7.3 Programming guidelines specific to master mode

This section presents the following topics:

- [Generating START](#)
- [Transmit/receive sequence](#)
- [Generating STOP](#)
- [Generating repeated START](#)
- [Loss of arbitration](#)

#### 26.7.3.1 Generating START

After completion of the initialization procedure, serial data can be transmitted by selecting the 'master transmitter' mode. If the device is connected to a multi-master bus system, the state of the I<sup>2</sup>C Bus Busy bit (IBB) must be tested to check whether the serial bus is free.

If the bus is free (IBB=0), the start condition and the first byte (the slave address) can be sent. The data written to the data register comprises the slave calling address and the LSB, which is set to indicate the direction of transfer required from the slave.

The bus free time (i.e., the time between a STOP condition and the following START condition) is built into the hardware that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period, it may be necessary to wait until the I<sup>2</sup>C is busy after writing the calling address to the IBDR before proceeding with the following instructions. This is illustrated in the following example.

An example of the sequence of events which generates the START signal and transmits the first byte of data (slave address) is shown below:

```

while (IBSR[IBB]==1)           // wait in loop for IBB flag to clear
IBCR[MS/MSL] and IBCR[]Tx/Rx] = 1 // master and transmit mode, that is,
// generate start condition
IBDR = calling_address          // send the calling address to the data register
while (bit 5, IBSR ==0)          // wait in loop for IBB flag to be set

```

### 26.7.3.2 Transmit/receive sequence

The following tables present the sequences for:

- Master transmit
- Master receive
- Slave transmit
- Slave receive

**Table 26-16. Master transmit sequence**

Step	Action
a	Use <a href="#">I2C Bus Frequency Divider Register (I2C_IBFD)</a> to select the required division ratio to obtain SCL frequency from Platform clock/2.
b	Write 0 to the IBDIS field in <a href="#">I2C Bus Control Register (I2C_IBCR)</a> to enable the I <sup>2</sup> C interface system.
c	Use <a href="#">I2C Bus Control Register (I2C_IBCR)</a> to select Master mode, Transmit mode, and interrupt enable.
d	Write 0 to the IBIF field in <a href="#">I2C Bus Status Register (I2C_IBSR)</a> .
e	Write data to <a href="#">I2C Bus Data I/O Register (I2C_IBDR)</a> .
f	Observe changes in the TCF field of <a href="#">I2C Bus Status Register (I2C_IBSR)</a> : <ul style="list-style-type: none"> <li>• When IBSR[TCF] becomes 0, the transfer is in progress.</li> <li>• When IBSR[TCF] becomes 1, the transfer is complete.</li> </ul>
g	Wait until the IBIF field in <a href="#">I2C Bus Status Register (I2C_IBSR)</a> becomes 1.
h	Read the fields in <a href="#">I2C Bus Status Register (I2C_IBSR)</a> to determine what happened: <ul style="list-style-type: none"> <li>• If TCF = 1, the transfer completed.</li> <li>• If RXAK = 1, a No Acknowledge condition occurred.</li> <li>• If IBB = 0, the bus transitioned from Busy to Idle state.</li> <li>• If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1).</li> </ul> <p><b>NOTE:</b> You can ignore Address Detect (IAAS = 1) for master mode (it is valid only for slave mode).</p>
i	Examine the RXAK field in <a href="#">I2C Bus Status Register (I2C_IBSR)</a> for an acknowledgment from the slave.
j	Repeat steps d through i to transfer the next consecutive bytes of data.

**Table 26-17. Master receive sequence**

Step	Action
a	Follow steps a through i in Table 26-16 for address dispatch.
b	Write 0 to the IBIF field in I <sup>2</sup> C Bus Status Register (I2C_IBSR).
c	Write 0 to the TXRX field in I <sup>2</sup> C Bus Control Register (I2C_IBCR) to select Receive mode.
d	Perform a dummy read of I <sup>2</sup> C Bus Data I/O Register (I2C_IBDR) to initiate the receive operation.
e	Wait until the TCF field in I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 1. (This proves that the transfer is complete.)
f	Wait until the IBIF field in I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 1.
g	Read the fields in I <sup>2</sup> C Bus Status Register (I2C_IBSR) to determine what happened: <ul style="list-style-type: none"> <li>If TCF = 1, the reception completed.</li> <li>If IBB = 0, the bus transitioned from Busy to Idle state.</li> <li>If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1).</li> </ul> <b>NOTE:</b> You can ignore the No Acknowledge condition (RXAK = 1) for receive mode.
h	Read I <sup>2</sup> C Bus Data I/O Register (I2C_IBDR) to determine the data received from the slave.

**Table 26-18. Slave transmit sequence**

Step	Action
a	Use I <sup>2</sup> C Bus Address Register (I2C_IBAD) to define the slave address.
b	Write 0 to the IBDIS field in I <sup>2</sup> C Bus Control Register (I2C_IBCR) to enable the I <sup>2</sup> C interface system.
c	Examine fields in I <sup>2</sup> C Bus Status Register (I2C_IBSR) as follows: <ul style="list-style-type: none"> <li>If IAAS = 1, examine IBSR[SRW].</li> <li>If IAAS = 1 and SRW = 1, write 1 to IBCR[TXRX] to select Transmit mode.</li> </ul>
d	Write data to I <sup>2</sup> C Bus Data I/O Register (I2C_IBDR).
e	Wait until the IBIF field in I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 1.
f	Wait until the RXAK field in I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 0.
g	Write 0 to the IBIF field in I <sup>2</sup> C Bus Status Register (I2C_IBSR).
h	Repeat steps d through g for the next consecutive data transfers.

**Table 26-19. Slave receive sequence**

Step	Action
a	Use I <sup>2</sup> C Bus Address Register (I2C_IBAD) to define the slave address.
b	Write 0 to the IBDIS field of I <sup>2</sup> C Bus Control Register (I2C_IBCR) to enable the I <sup>2</sup> C interface system.
c	Examine fields in I <sup>2</sup> C Bus Status Register (I2C_IBSR) as follows: <ul style="list-style-type: none"> <li>If IAAS = 1, examine IBSR[SRW].</li> <li>If IAAS = 1 and SRW = 0, write 0 to IBCR[TXRX] to select Receive mode.</li> </ul>
d	Write 0 to the IBIF field of I <sup>2</sup> C Bus Status Register (I2C_IBSR).
e	Perform a dummy read of I <sup>2</sup> C Bus Data I/O Register (I2C_IBDR) to initiate the receive operation.
f	Wait until the TCF field of I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 1. (This proves that the transfer is complete.)
g	Wait until the IBIF field of I <sup>2</sup> C Bus Status Register (I2C_IBSR) becomes 1.

Table continues on the next page...

**Table 26-19. Slave receive sequence (continued)**

Step	Action
h	<p>Read the fields in <a href="#">I2C Bus Status Register (I2C_IBSR)</a> to determine what happened:</p> <ul style="list-style-type: none"> <li>• If TCF = 1, the reception completed.</li> <li>• If IBB = 0, the bus transitioned from Busy to Idle state.</li> <li>• If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1).</li> </ul> <p><b>NOTE:</b> You can ignore the No Acknowledge condition (RXAK = 1) for receive mode.</p>
i	Read <a href="#">I2C Bus Data I/O Register (I2C_IBDR)</a> to determine the data received from the master.

### 26.7.3.3 Generating STOP

A data transfer ends with a STOP signal generated by the 'master' device. A master transmitter can simply generate a STOP signal after all the data has been transmitted. The following is an example showing how a STOP condition is generated by a master transmitter.

```
if (tx_count == 0) or      // check to see if all data bytes have been transmitted
  (bit_0, IBSR == 1) {    // or if no ACK generated
    clear bit 5, IBCR    // generate stop condition
  }
else {
  IBDR = data_to_transmit // write byte of data to DATA register
  tx_count --             // decrement counter
}                         // return from interrupt
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data which can be done by setting the NOACK bit in IBCR before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must first be generated. The following is an example showing how a STOP signal is generated by a master receiver.

```
rx_count --                // decrease the rx counter
if (rx_count == 1)         // 2nd last byte to be read ?
  bit 3, IBCR = 1         // disable ACK
  if (rx_count == 0)       // last byte to be read ?
    bit 5, IBCR = 0         // generate stop signal
else
  data_received = IBDR   // read RX data and store
```

### 26.7.3.4 Generating repeated START

At the end of data transfer, if the master still wants to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

```

bit 2, IBCR = 1           // generate another start (restart)
IBDR == calling_address   // transmit the calling address

```

### 26.7.3.5 Loss of arbitration

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices that lost arbitration are immediately switched to slave mode by the hardware. Their data output to the SDA line is stopped, but SCL is still generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with IBAL=1 and MS/SL=0. If one master attempts to start transmission, while the bus is being engaged by another master, the hardware will inhibit the transmission, switch the MS/SL bit from 1 to 0 without generating a STOP condition, generate an interrupt to CPU, set the IBAL to indicate that the attempt to engage the bus is failed, and not set the TCF due to the loss of data during arbitration. When considering these cases, the slave service routine should test the IBAL first and the software should clear the IBAL bit if it is set.

### 26.7.4 Programming guidelines specific to slave mode

In the slave interrupt service routine, the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the transmit/receive mode select bit (Tx/Rx bit of IBCR) according to the R/W command bit (SRW). Writing to the IBCR clears IAAS automatically. Note that the only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred. Interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer may now be initiated by writing information to IBDR for slave transmits or dummy reading from IBDR in slave receive mode. The slave will drive SCL low in-between byte transfers, SCL is released when the IBDR is accessed in the required mode.

In slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. Setting RXAK means an "end of data" signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

## 26.7.5 DMA application information

The DMA interface on the I<sup>2</sup>C is not completely autonomous and requires intervention from the CPU to start and to terminate the frame transfer. DMA mode is only valid for Master transmit and Master receive modes. Software must ensure that the DMAEN field in [I2C Bus Control Register \(I2C\\_IBCR\)](#) is not set when the I<sup>2</sup>C module is configured in slave mode.

The DMA controller must only transfer one byte of data per Tx/Rx request. This is because there is no FIFO on the I<sup>2</sup>C block.

The CPU should also keep the I<sup>2</sup>C interrupt enabled during a DMA transfer to detect the arbitration lost condition and take action to recover from this situation. The DMAEN field in [I2C Bus Control Register \(I2C\\_IBCR\)](#) works as a disable for the transfer complete interrupt. This means that during normal transfers (no errors) there will always be either an interrupt or a request to the DMA controller, depending on the setting of the DMAEN field. All error conditions will trigger an interrupt and require CPU intervention. The address match condition will not occur in DMA mode as the I<sup>2</sup>C should never be configured for slave operation.

The following sections detail how to set up a DMA transfer and what intervention is required from the CPU. It is assumed that the system DMA controller is capable of generating an interrupt after a certain number of DMA transfers have taken place.

The sections present the following topics:

- [DMA mode, master transmit](#)
- [DMA mode, master reception](#)
- [Exiting DMA mode, system requirement considerations](#)

### 26.7.5.1 DMA mode, master transmit

The following flow diagram details exactly the operation for using a DMA controller to transmit "n" data bytes to a slave. The first byte (the slave calling address) is always transmitted by the CPU. All subsequent data bytes (apart from the last data byte) can be transferred by the DMA controller. The last data byte must be transferred by the CPU.

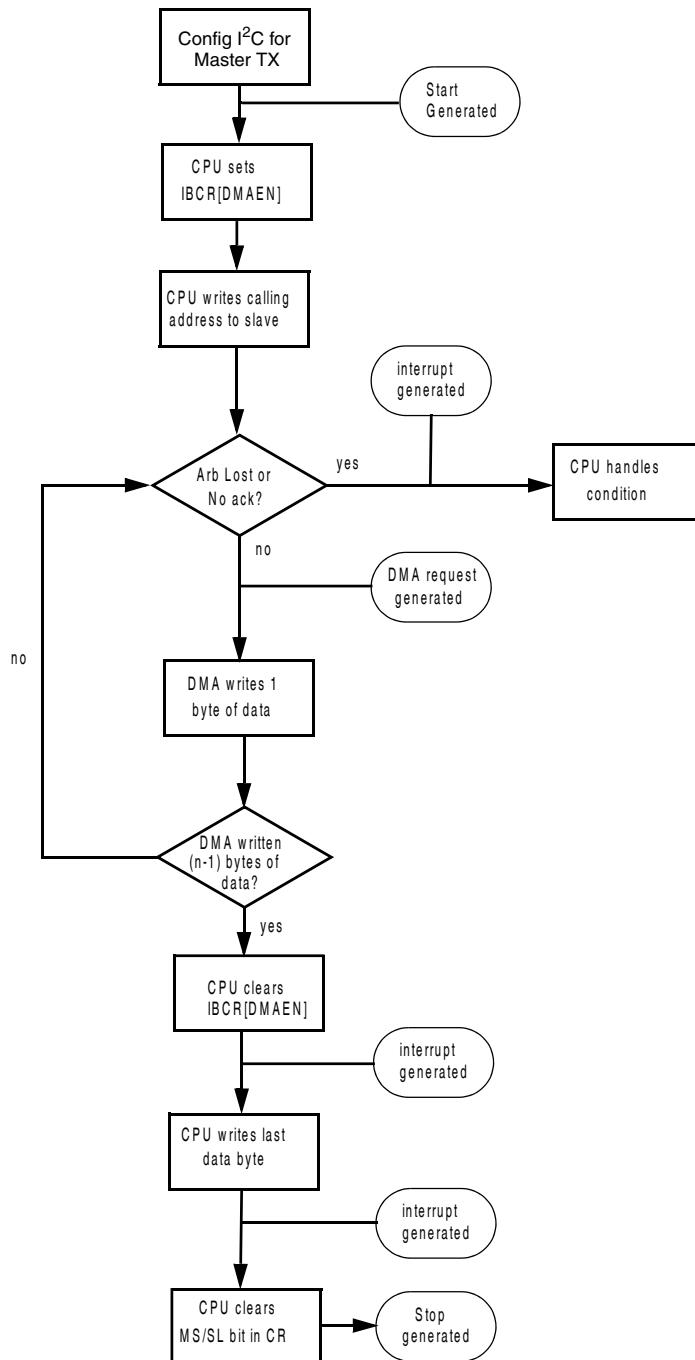
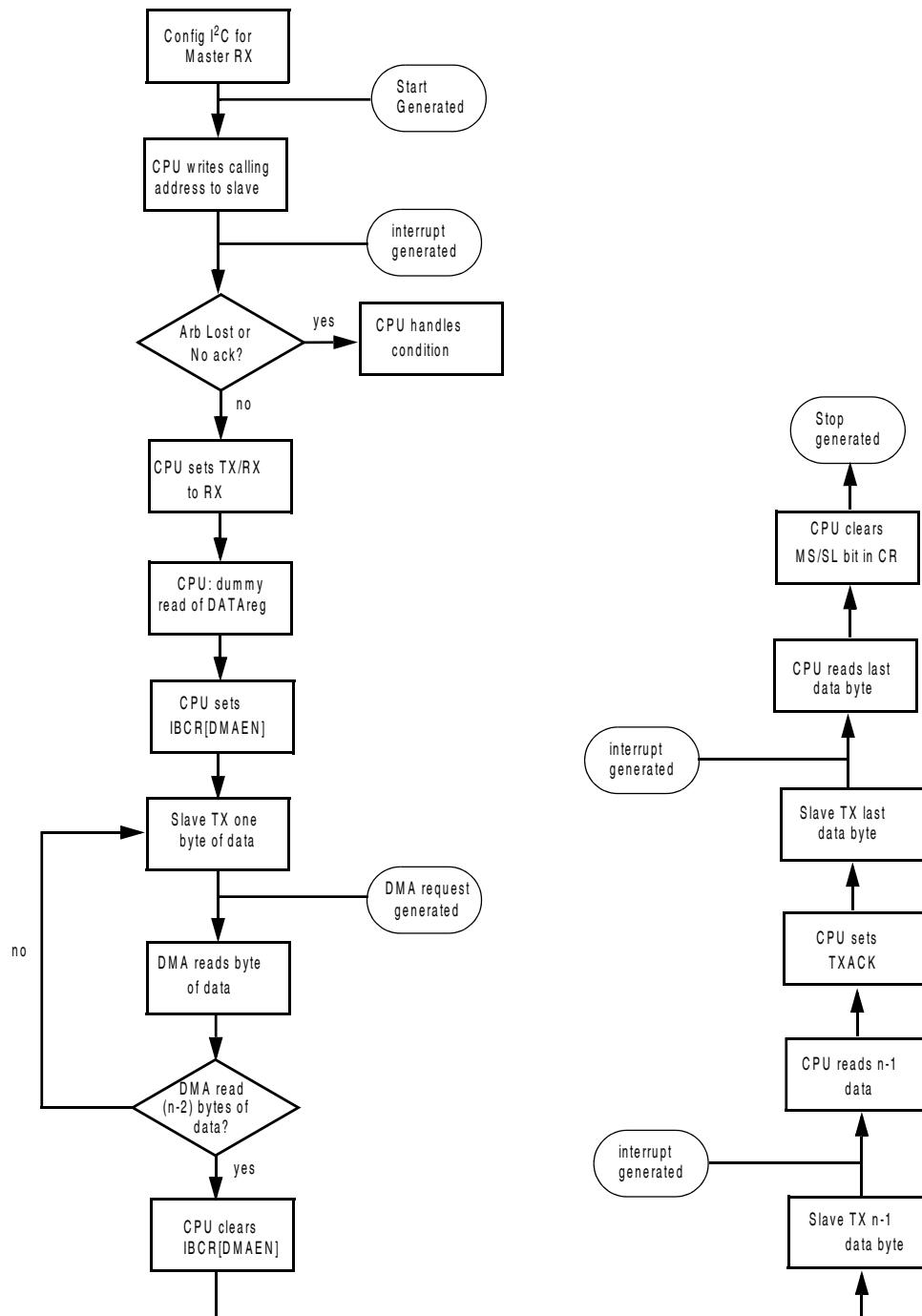


Figure 26-10. Flow-Chart of DMA mode master transmit

### 26.7.5.2 DMA mode, master reception

The following flow diagram details the exact operation for using a DMA controller to receive "n" data bytes from a slave. The first byte (the slave calling address) is always transmitted by the CPU. All subsequent data bytes (apart from the two last data bytes) can be read by the DMA controller. The last two data bytes must be transferred by the CPU.



**Figure 26-11. Flow-Chart of DMA mode master receive**

### 26.7.5.3 Exiting DMA mode, system requirement considerations

As described above, the final transfers of both TX and RX transfers need to be handled via interrupt by the CPU. To change from DMA to interrupt driven transfers in the I<sup>2</sup>C module, you have to disable the DMAEN bit in the IBCR register. The trigger to exit the DMA mode is that the programmed DMA Transfer Control Descriptor (TCD) has completed all its transfers to/from the I<sup>2</sup>C module.

After the last DMA write (TX mode) to the I<sup>2</sup>C the module will immediately start the next I<sup>2</sup>C-bus transfer. The same is true for receive mode. After the DMA read from the IBDR register the module initiates the next I<sup>2</sup>C-bus transfer. This results in two possible scenarios in the DMA mode exiting scheme.

#### 1. Fast reaction

The DMAEN bit is cleared before the next I<sup>2</sup>C-bus transfer completes. In this case the module will raise an interrupt request to the CPU which can be serviced normally.

#### 2. Slow reaction

The DMAEN bit is cleared after the next I<sup>2</sup>C-bus transfer has already completed. In this case, the module will not raise an interrupt request to the CPU. Instead the TCF bit can be read to determine that the transfer completed and the module is ready for further transfer.

What is fast/slow reaction?

The reaction time  $T_R$  for the system to disable DMAEN after the last DMA controller access to the I<sup>2</sup>C is the time required for one byte transfer over the I<sup>2</sup>C. For 'fast reaction' the disabling has to occur before the 9<sup>th</sup> bit of the data transfer which is the ACK bit. So the time available is eight times the SCL period.

$$T_R = 8 \times T_{SCL}$$

In fast mode, with 400kbit/s,  $T_{SCL}$  is 2.5μs, so  $T_R$  is 20μs.

Depending on the system and DMA controller there are different possibilities for the de-assertion of DMAEN. Three options are:

#### 1. CPU intervention via Interrupt

The DMA controller is programmed to signal an interrupt to the CPU which is then responsible for the de-assertion of DMAEN. This scheme should be supported by most systems but it can result in a slow reaction time if other higher priority

interrupts interfere. Therefore the interrupt handling routine can become complicated as it has to check which of the two cases happened (check TCF bit) and act accordingly. In case of slow reaction you can force an interrupt for the I<sup>2</sup>C in the interrupt controller to have the further transfer handled by the normal I<sup>2</sup>C interrupt routine.

**Note**

The use of nested interrupts can still cause potential issues in this scenario, if someone tries to stall the DMA interrupt between the de-assertion and DMAEN bit and checks the TCF bit.

## 2. DMA channel linking

If the Transfer control descriptor in the DMA controller that performs the data transfer is linked to another channel that does a write to IBCR to disable the DMAEN field, this might probably be the fastest system solution, but it uses two DMA channels.

**Note**

Here you have to make sure on system level that no higher priority DMA requests occur between the two linked TCDs as those could again create a scenario of slow reaction.

## 3. DMA scatter/gather process

If the Transfer control descriptor in the DMA controller that performs the data transfer has the scatter/gather feature activated, this feature will initiate a reload of another TCD from system RAM after the completion of the first TCD. The new TCD will have its start bit already set and immediately start the required write to the IBCR to disable the DMAEN field. This TCD also has scatter/gather activated and is programmed to reload the initial TCD upon completion, bringing the system back into a "ready-for-I<sup>2</sup>C-transfer" state. The advantage over the two other solutions is that this requires neither CPU intervention nor a second DMA channel. This comes at the cost of 64 bytes RAM (two TCDs), some system bus transfer overhead and a little increase in application code complexity.

## Note

Here you have to make sure at system level that no higher priority DMA requests occur during the scatter/gather process, as those could again create a scenario of slow reaction.

Example latencies for a 32 MHz system with a full speed 32-bit AHB bus and an I<sup>2</sup>C connected via half speed IPI bus:

- Accessing the I<sup>2</sup>C from the DMA controller via IPI bus typically requires four cycles (consecutive accesses to the I<sup>2</sup>C could be faster):

$$4 \times T_{\text{IPI}} = 4/16 \text{ MHz} = 250 \text{ ns}$$

- Reloading a new TCD (8 x 32-bit) via AHB to the DMA controller (scatter/gather process):

$$8 \times T_{\text{AHD}} = 8/32 \text{ MHz} = 250 \text{ ns}$$

Example latencies for a 150 MHz system with a full speed 32-bit AHB bus and an I<sup>2</sup>C connected via half speed IPI bus:

- Accessing the I<sup>2</sup>C from the DMA controller via IPI bus typically requires four cycles (consecutive accesses to the I<sup>2</sup>C could be faster):

$$4 \times T_{\text{IPI}} = 4/150 \text{ MHz} = 26.6 \text{ ns}$$

- Reloading a new TCD (4 x 64-bit) via AHB to the DMA controller (scatter/gather process):

$$4 \times T_{\text{AHD}} = 4/150 \text{ MHz} = 26.6 \text{ ns}$$

With the DMA scatter/gather process the required IBCR access can be done in 0.5 µs, leaving a large margin of 19.5 µs for additional system delays. In this way, the slow reaction case can be prevented. The system user needs to decide which usage model best suits his overall requirement.



# **Chapter 27**

## **Low Power Universal asynchronous receiver/transmitter (LPUART)**

### **27.1 LPUART module integration**

The following table describes the LPUART module integration into the chip:

**Table 27-1. LPUART module integration**

Module	Module Base address
LPUART1	295_0000
LPUART2	296_0000
LPUART3	297_0000
LPUART4	298_0000
LPUART5	299_0000
LPUART6	29A_0000

Additionally:

- All modules have been integrated with identical parameters.
- The hardware flow control (CTS/RTS) is supported only on LPUART1, LPUART2, and LPUART3.
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format is supported only on LPUART1, LPUART2, and LPUART3.
- All LPUART interrupts from a single LPUART module are ORed together before feeding to the GIC interrupt controller. Therefore, GIC has one interrupt port for each LPUART instance.

The remainder of this chapter refers to a single LPUART module. Notes are included to indicate variations for multiple instantiations.

## 27.2 Chip LPUART signals

The following table lists the SoC signal names and their corresponding LPUART module signal names used in this chapter:

**Table 27-2. LPUART signals**

Signal name	LPUART module signal
LPUART $n$ _SOUT	LPUART_TX
LPUART $n$ _SIN	LPUART_RX
LPUART $n$ _RTS_B	LPUART_RTS
LPUART $n$ _CTS_B	LPUART_CTS

## 27.3 Chip LPUART module special consideration

The LPUART module implements the following parameter settings in the chip:

**Table 27-3. LPUART parameter settings**

LPUART parameters	LS1043A parameter value
DATA_WD	32
TXFIFO_SZ	4
RXFIFO_SZ	4
Stop mode support	Yes. Refers to the LPM20 low power mode of the chip.
Doze mode support	No

The following table provides the clock source for each LPUART module:

**Table 27-4. LPUART clocking**

LPUART module	Clock source
LPUART1	SYS_REF_CLK
LPUART2	platform clock/2
LPUART3	platform clock/2
LPUART4	platform clock/2
LPUART5	platform clock/2
LPUART6	platform clock/2

## 27.4 Introduction

### 27.4.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
- Optional 13-bit break character generation / 11-bit break character detection
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- Independent FIFO structure for transmit and receive
  - Separate configurable watermark for receive and transmit requests

## 27.4.2 Modes of operation

### 27.4.2.1 Stop mode

The LPUART will remain functional during Stop mode, provided the asynchronous transmit and receive clock remains enabled. The LPUART can generate an interrupt or DMA request to cause a wakeup from Stop mode.

### 27.4.2.2 Wait mode

The LPUART can be configured to Stop in Wait modes, when the DOZEEN bit is set. The transmitter and receiver will finish transmitting/receiving the current word.

## 27.4.3 Signal Descriptions

Signal	Description	I/O
LPUART_TX	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is disabled or transmit direction is configured for receive data.	I/O
LPUART_RX	Receive data.	I
LPUART_CTS	Clear to send.	I
LPUART_RTS	Request to send.	O

## 27.4.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

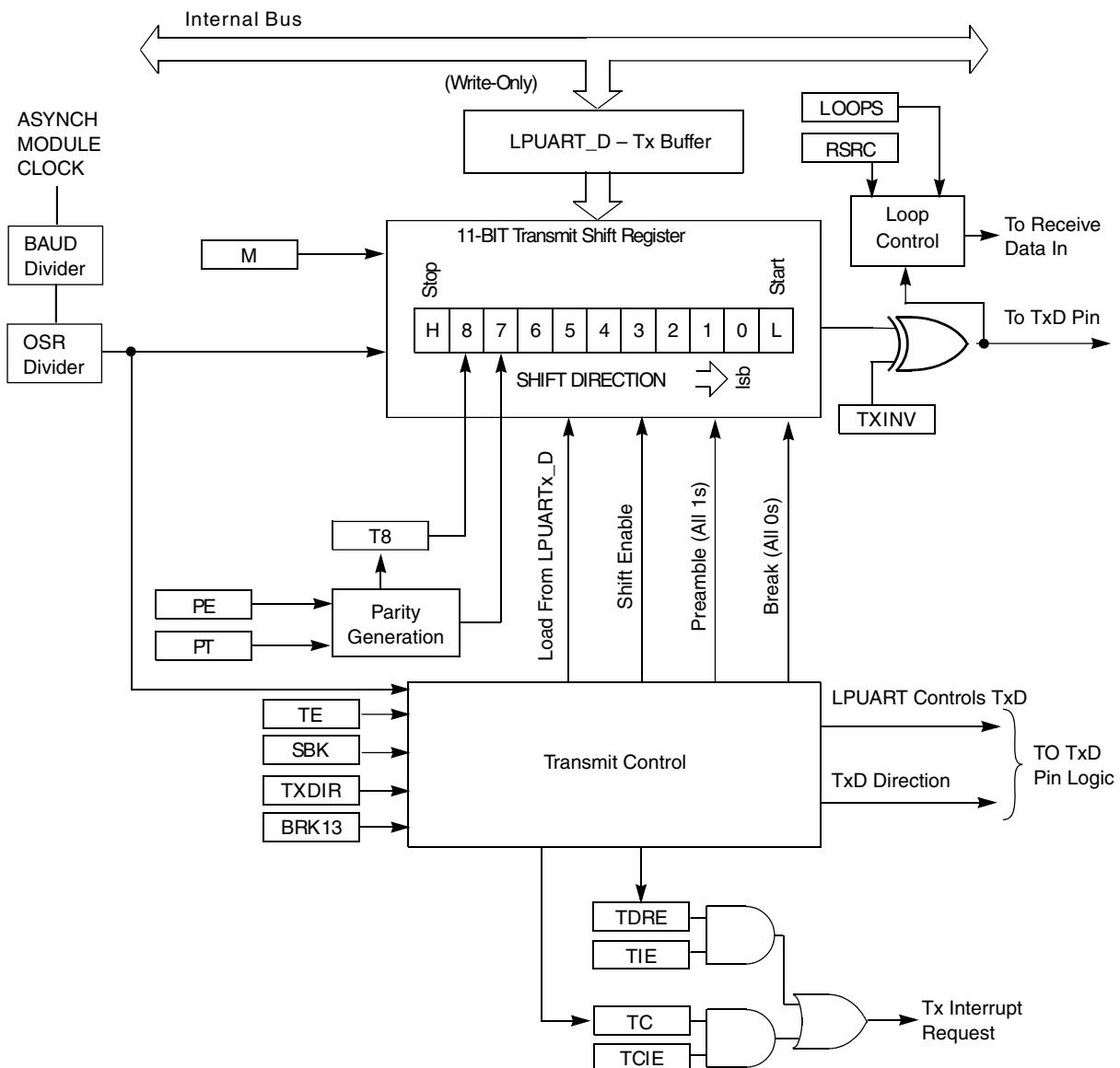
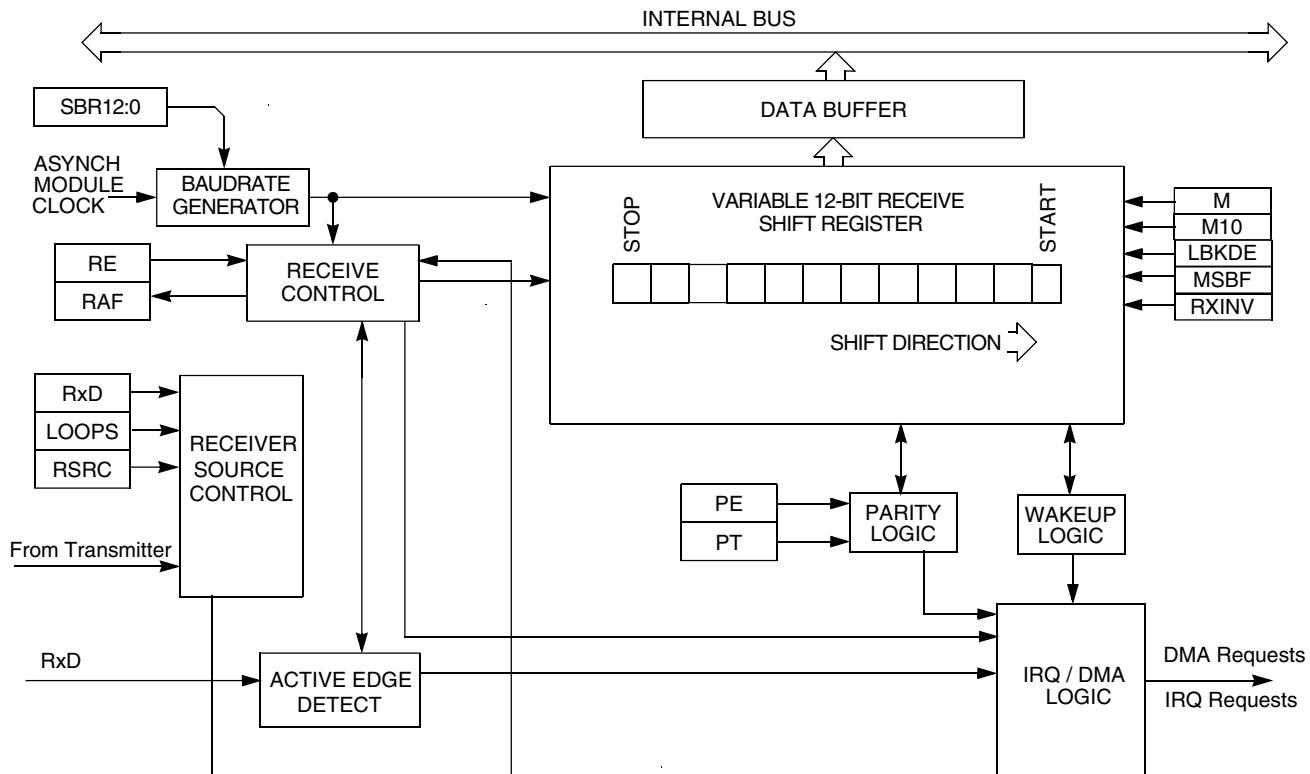


Figure 27-1. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

## Register definition



**Figure 27-2. LPUART receiver block diagram**

## 27.5 Register definition

The LPUART includes registers to control baud rate, select LPUART options, report LPUART status, and for transmit/receive data. Access to an address outside the valid memory map will generate a bus error.

**LPUART memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
295_0000	LPUART Baud Rate Register (LPUART1_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>
295_0004	LPUART Status Register (LPUART1_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
295_0008	LPUART Control Register (LPUART1_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
295_000C	LPUART Data Register (LPUART1_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
295_0010	LPUART Match Address Register (LPUART1_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
295_0014	LPUART Modem IrDA Register (LPUART1_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
295_0018	LPUART FIFO Register (LPUART1_FIFO)	32	R/W	See section	<a href="#">27.5.7/1462</a>
295_001C	LPUART Watermark Register (LPUART1_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>
296_0000	LPUART Baud Rate Register (LPUART2_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>

*Table continues on the next page...*

**LPUART memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
296_0004	LPUART Status Register (LPUART2_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
296_0008	LPUART Control Register (LPUART2_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
296_000C	LPUART Data Register (LPUART2_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
296_0010	LPUART Match Address Register (LPUART2_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
296_0014	LPUART Modem IrDA Register (LPUART2_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
296_0018	LPUART FIFO Register (LPUART2_FIFO)	32	R/W	<a href="#">See section</a>	<a href="#">27.5.7/1462</a>
296_001C	LPUART Watermark Register (LPUART2_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>
297_0000	LPUART Baud Rate Register (LPUART3_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>
297_0004	LPUART Status Register (LPUART3_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
297_0008	LPUART Control Register (LPUART3_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
297_000C	LPUART Data Register (LPUART3_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
297_0010	LPUART Match Address Register (LPUART3_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
297_0014	LPUART Modem IrDA Register (LPUART3_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
297_0018	LPUART FIFO Register (LPUART3_FIFO)	32	R/W	<a href="#">See section</a>	<a href="#">27.5.7/1462</a>
297_001C	LPUART Watermark Register (LPUART3_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>
298_0000	LPUART Baud Rate Register (LPUART4_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>
298_0004	LPUART Status Register (LPUART4_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
298_0008	LPUART Control Register (LPUART4_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
298_000C	LPUART Data Register (LPUART4_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
298_0010	LPUART Match Address Register (LPUART4_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
298_0014	LPUART Modem IrDA Register (LPUART4_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
298_0018	LPUART FIFO Register (LPUART4_FIFO)	32	R/W	<a href="#">See section</a>	<a href="#">27.5.7/1462</a>
298_001C	LPUART Watermark Register (LPUART4_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>
299_0000	LPUART Baud Rate Register (LPUART5_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>
299_0004	LPUART Status Register (LPUART5_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
299_0008	LPUART Control Register (LPUART5_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
299_000C	LPUART Data Register (LPUART5_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
299_0010	LPUART Match Address Register (LPUART5_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
299_0014	LPUART Modem IrDA Register (LPUART5_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
299_0018	LPUART FIFO Register (LPUART5_FIFO)	32	R/W	<a href="#">See section</a>	<a href="#">27.5.7/1462</a>
299_001C	LPUART Watermark Register (LPUART5_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>
29A_0000	LPUART Baud Rate Register (LPUART6_BAUD)	32	R/W	0F00_0004h	<a href="#">27.5.1/1448</a>
29A_0004	LPUART Status Register (LPUART6_STAT)	32	R/W	00C0_0000h	<a href="#">27.5.2/1450</a>
29A_0008	LPUART Control Register (LPUART6_CTRL)	32	R/W	0000_0000h	<a href="#">27.5.3/1454</a>
29A_000C	LPUART Data Register (LPUART6_DATA)	32	R/W	0000_0000h	<a href="#">27.5.4/1458</a>
29A_0010	LPUART Match Address Register (LPUART6_MATCH)	32	R/W	0000_0000h	<a href="#">27.5.5/1459</a>
29A_0014	LPUART Modem IrDA Register (LPUART6_MODIR)	32	R/W	0000_0000h	<a href="#">27.5.6/1460</a>
29A_0018	LPUART FIFO Register (LPUART6_FIFO)	32	R/W	<a href="#">See section</a>	<a href="#">27.5.7/1462</a>

Table continues on the next page...

## LPUART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
29A_001C	LPUART Watermark Register (LPUART6_WATER)	32	R/W	0000_0000h	<a href="#">27.5.8/1464</a>

**27.5.1 LPUART Baud Rate Register (LPUARTx\_BAUD)**

Address: Base address + 0h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MAEN1	MAEN2	M10	OSR				TDMAE	0	RDMAE	0	0	BOTHEDGE	RESYNCDIS		
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	LBKDIIE	RXEDGEIE	SBNS	SBR												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

**LPUARTx\_BAUD field descriptions**

Field	Description
0 MAEN1	Match Address Mode Enable 1 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA1].
1 MAEN2	Match Address Mode Enable 2 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA2].
2 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled. 0 Receiver and transmitter use 8-bit or 9-bit data characters. 1 Receiver and transmitter use 10-bit data characters.
3–7 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver between 4x (00011) and 32x (11111). Writing an invalid oversampling ratio (for example, a value not between 4x and 32x) will default to an oversampling ratio of 16 (01111). The OSR field should only be changed when the transmitter and receiver are both disabled. Note that the oversampling ratio = OSR + 1.

Table continues on the next page...

**LPUARTx\_BAUD field descriptions (continued)**

Field	Description
8 TDMAE	<p>Transmitter DMA Enable</p> <p>TDMAE configures the transmit data register empty flag, LPUART_STAT[TDRE], to generate a DMA request.</p> <p>0 DMA request disabled. 1 DMA request enabled.</p>
9 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
10 RDMAE	<p>Receiver Full DMA Enable</p> <p>RDMAE configures the receiver data register full flag, LPUART_STAT[RDRF], to generate a DMA request.</p> <p>0 DMA request disabled. 1 DMA request enabled.</p>
11 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
12–13 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
14 BOTHEDGE	<p>Both Edge Sampling</p> <p>Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled.</p> <p>0 Receiver samples input data using the rising edge of the baud rate clock. 1 Receiver samples input data using the rising and falling edge of the baud rate clock.</p>
15 RESYNCDIS	<p>Resynchronization Disable</p> <p>When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled.</p> <p>0 Resynchronization during received data word is supported 1 Resynchronization during received data word is disabled</p>
16 LBKDIE	<p>LIN Break Detect Interrupt Enable</p> <p>LBKDIE enables the LIN break detect flag, LBKDIF, to generate interrupt requests.</p> <p>0 Hardware interrupts from LPUART_STAT[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[LBKDIF] flag is 1.</p>
17 RXEDGIE	<p>RX Input Active Edge Interrupt Enable</p> <p>Enables the receive input active edge, RXEDGIF, to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the RXEDGIF to set.</p> <p>0 Hardware interrupts from LPUART_STAT[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[RXEDGIF] flag is 1.</p>
18 SBNS	<p>Stop Bit Number Select</p> <p>SBNS determines whether data characters are one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled.</p>

*Table continues on the next page...*

## Register definition

### LPUARTx\_BAUD field descriptions (continued)

Field	Description
	0 One stop bit. 1 Two stop bits.
19–31 SBR	Baud Rate Modulo Divisor.  The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) × SBR)". The 13-bit baud rate setting [SBR12:SBR0] must only be updated when the transmitter and receiver are both disabled (LPUART_CTRL[RE] and LPUART_CTRL[TE] are both 0).

## 27.5.2 LPUART Status Register (LPUARTx\_STAT)

Address: Base address + 4h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	LBKDIF	RXEDGIF		MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	w1c	w1c											w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R									0								
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### LPUARTx\_STAT field descriptions

Field	Description
0 LBKDIF	LIN Break Detect Interrupt Flag  LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it.

Table continues on the next page...

**LPUARTx\_STAT field descriptions (continued)**

Field	Description
	<p>0 No LIN break character has been detected. 1 LIN break character has been detected.</p>
1 RXEDGIF	<p>LPUART_RX Pin Active Edge Interrupt Flag  RXEDGIF is set whenever the receiver is enabled and an active edge, falling if RXINV = 0, rising if RXINV=1, on the LPUART_RX pin occurs. RXEDGIF is cleared by writing a 1 to it.</p> <p>0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.</p>
2 MSBF	<p>MSB First  Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled.</p> <p>0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1 MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE].</p>
3 RXINV	<p>Receive Data Inversion  Setting this bit reverses the polarity of the received data input.  <b>NOTE:</b> Setting RXINV inverts the LPUART_RX input for all cases: data bits, start and stop bits, break, and idle.</p> <p>0 Receive data not inverted. 1 Receive data inverted.</p>
4 RWUID	<p>Receive Wake Up Idle Detect  For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled.</p> <p>0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not get set when an address does not match. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does get set when an address does not match.</p>
5 BRK13	<p>Break Character Generation Length  BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled.</p> <p>0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1). 1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 16 (if M10 = 1, SNBS = 1).</p>
6 LBKDE	<p>LIN Break Detection Enable  LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer.</p>

*Table continues on the next page...*

**LPUARTx\_STAT field descriptions (continued)**

Field	Description
0	Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1).
1	Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 14 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 15 (if M10 = 1, SNBS = 1).
7 RAF	<p>Receiver Active Flag</p> <p>RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line.</p> <p>0 LPUART receiver idle waiting for a start bit. 1 LPUART receiver active (LPUART_RX input not idle).</p>
8 TDRE	<p>Transmit Data Register Empty Flag</p> <p>When the transmit FIFO is enabled, TDRE will set when the number of datawords in the transmit FIFO (LPUART_DATA) is equal to or less than the number indicated by LPUART_WATER[TXWATER]). To clear TDRE, write to the LPUART data register (LPUART_DATA) until the number of words in the transmit FIFO is greater than the number indicated by LPUART_WATER[TXWATER]. When the transmit FIFO is disabled, TDRE will set when the transmit data register (LPUART_DATA) is empty. To clear TDRE, write to the LPUART data register (LPUART_DATA).</p> <p>TDRE is not affected by a character that is in the process of being transmitted, it is updated at the start of each transmitted character.</p> <p>0 Transmit data buffer full. 1 Transmit data buffer empty.</p>
9 TC	<p>Transmission Complete Flag</p> <p>TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to LPUART_DATA to transmit new data, queuing a preamble by clearing and then setting LPUART_CTRL[TE], queuing a break character by writing 1 to LPUART_CTRL[SBK].</p> <p>0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).</p>
10 RDRF	<p>Receive Data Register Full Flag</p> <p>When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by LPUART_WATER[RXWATER]. To clear RDRF, read LPUART_DATA until the number of datawords in the receive data buffer is equal to or less than the number indicated by LPUART_WATER[RXWATER]. When the receive FIFO is disabled, RDRF is set when the receive buffer (LPUART_DATA) is full. To clear RDRF, read the LPUART_DATA register.</p> <p>A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received.</p> <p>0 Receive data buffer empty. 1 Receive data buffer full.</p>
11 IDLE	<p>Idle Line Flag</p> <p>IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting</p>

*Table continues on the next page...*

**LPUARTx\_STAT field descriptions (continued)**

Field	Description
	<p>idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line.</p> <p>To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot become set again until after a new character has been stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period.</p> <p>0 No idle line detected. 1 Idle line was detected.</p>
12 OR	<p>Receiver Overrun Flag</p> <p>OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received.</p> <p>While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag.</p> <p>0 No overrun. 1 Receive overrun (new LPUART data lost).</p>
13 NF	<p>Noise Flag</p> <p>The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from LPUART_DATA was received with noise detected within the character. To clear NF, write logic one to the NF.</p> <p>0 No noise detected. 1 Noise detected in the received character in LPUART_DATA.</p>
14 FE	<p>Framing Error Flag</p> <p>FE is set whenever the next character to be read from LPUART_DATA was received with logic 0 detected where a stop bit was expected. To clear FE, write logic one to the FE.</p> <p>0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error.</p>
15 PF	<p>Parity Error Flag</p> <p>PF is set whenever the next character to be read from LPUART_DATA was received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic one to the PF.</p> <p>0 No parity error. 1 Parity error.</p>
16–31 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>

### 27.5.3 LPUART Control Register (LPUARTx\_CTRL)

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

Address: Base address + 8h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				0					LOOPS	DOZEEN	RSR C	M	WAKE	ILT	PE	PT
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### LPUARTx\_CTRL field descriptions

Field	Description
0 R8T9	<p>Receive Bit 8 / Transmit Bit 9</p> <p>R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading LPUART_DATA.</p> <p>T9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing LPUART_DATA. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.</p>
1 R9T8	<p>Receive Bit 9 / Transmit Bit 8</p> <p>R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading LPUART_DATA</p> <p>T8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing LPUART_DATA. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.</p>
2 TXDIR	<p>LPUART_TX Pin Direction in Single-Wire Mode</p> <p>When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the LPUART_TX pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the LPUART_TX pin.</p> <p>0 LPUART_TX pin is an input in single-wire mode. 1 LPUART_TX pin is an output in single-wire mode.</p>

Table continues on the next page...

**LPUARTx\_CTRL field descriptions (continued)**

Field	Description
3 TXINV	<p>Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output.</p> <p><b>NOTE:</b> Setting TXINV inverts the LPUART_TX output for all cases: data bits, start and stop bits, break, and idle.</p> <ul style="list-style-type: none"> <li>0 Transmit data not inverted.</li> <li>1 Transmit data inverted.</li> </ul>
4 ORIE	<p>Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> <li>0 OR interrupts disabled; use polling.</li> <li>1 Hardware interrupt requested when OR is set.</li> </ul>
5 NEIE	<p>Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> <li>0 NF interrupts disabled; use polling.</li> <li>1 Hardware interrupt requested when NF is set.</li> </ul>
6 FEIE	<p>Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> <li>0 FE interrupts disabled; use polling.</li> <li>1 Hardware interrupt requested when FE is set.</li> </ul>
7 PEIE	<p>Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> <li>0 PF interrupts disabled; use polling).</li> <li>1 Hardware interrupt requested when PF is set.</li> </ul>
8 TIE	<p>Transmit Interrupt Enable Enables STAT[TDRE] to generate interrupt requests.</p> <ul style="list-style-type: none"> <li>0 Hardware interrupts from TDRE disabled; use polling.</li> <li>1 Hardware interrupt requested when TDRE flag is 1.</li> </ul>
9 TCIE	<p>Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests.</p> <ul style="list-style-type: none"> <li>0 Hardware interrupts from TC disabled; use polling.</li> <li>1 Hardware interrupt requested when TC flag is 1.</li> </ul>
10 RIE	<p>Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests.</p> <ul style="list-style-type: none"> <li>0 Hardware interrupts from RDRF disabled; use polling.</li> <li>1 Hardware interrupt requested when RDRF flag is 1.</li> </ul>

*Table continues on the next page...*

**LPUARTx\_CTRL field descriptions (continued)**

Field	Description
11 ILIE	<p>Idle Line Interrupt Enable</p> <p>ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests.</p> <p>0 Hardware interrupts from IDLE disabled; use polling. 1 Hardware interrupt requested when IDLE flag is 1.</p>
12 TE	<p>Transmitter Enable</p> <p>Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the LPUART_TX pin is tristated.</p> <p>0 Transmitter disabled. 1 Transmitter enabled.</p>
13 RE	<p>Receiver Enable</p> <p>Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any).</p> <p>0 Receiver disabled. 1 Receiver enabled.</p>
14 RWU	<p>Receiver Wakeup Control</p> <p>This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear.</p> <p><b>NOTE:</b> RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted.</p> <p>0 Normal receiver operation. 1 LPUART receiver in standby waiting for wakeup condition.</p>
15 SBK	<p>Send Break</p> <p>Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 to 13, or 13 to 16 if LPUART_STATBRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK.</p> <p>0 Normal transmitter operation. 1 Queue break character(s) to be sent.</p>
16–23 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
24 LOOPS	<p>Loop Mode Select</p> <p>When LOOPS is set, the LPUART_RX pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function.</p>

*Table continues on the next page...*

**LPUARTx\_CTRL field descriptions (continued)**

Field	Description
	<p>0 Normal operation - LPUART_RX and LPUART_TX use separate pins.</p> <p>1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).</p>
25 DOZEEN	<p>Doze Enable</p> <p>0 LPUART is enabled in Doze mode.</p> <p>1 LPUART is disabled in Doze mode.</p>
26 RSRC	<p>Receiver Source Select</p> <p>This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input.</p> <p>0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the LPUART_RX pin.</p> <p>1 Single-wire LPUART mode where the LPUART_TX pin is connected to the transmitter output and receiver input.</p>
27 M	<p>9-Bit or 8-Bit Mode Select</p> <p>0 Receiver and transmitter use 8-bit data characters.</p> <p>1 Receiver and transmitter use 9-bit data characters.</p>
28 WAKE	<p>Receiver Wakeup Method Select</p> <p>Determines which condition wakes the LPUART when RWU=1:</p> <ul style="list-style-type: none"> <li>Address mark in the most significant bit position of a received data character, or</li> <li>An idle condition on the receive pin input signal.</li> </ul> <p>0 Configures RWU for idle-line wakeup.</p> <p>1 Configures RWU with address-mark wakeup.</p>
29 ILT	<p>Idle Line Type Select</p> <p>Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions.</p> <p><b>NOTE:</b> In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count.</p> <p>0 Idle character bit count starts after start bit.</p> <p>1 Idle character bit count starts after stop bit.</p>
30 PE	<p>Parity Enable</p> <p>Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit.</p> <p>0 No hardware parity generation or checking.</p> <p>1 Parity enabled.</p>
31 PT	<p>Parity Type</p> <p>Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even.</p>

*Table continues on the next page...*

**LPUARTx\_CTRL field descriptions (continued)**

Field	Description
0	Even parity.
1	Odd parity.

**27.5.4 LPUART Data Register (LPUARTx\_DATA)**

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags.

Address: Base address + Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	NOISY	PARITYE	FRETSC		0		R9T9	R8T8	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LPUARTx\_DATA field descriptions**

Field	Description
0–15 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
16 NOISY	The current received dataword contained in DATA[R9:R0] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.
17 PARITYE	The current received dataword contained in DATA[R9:R0] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
18 FRETSC	Frame Error The current received dataword contained in DATA[R9:R0] was received with a frame error.
19–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 R9T9	Read receive data buffer 9 or write transmit data buffer 9.
23 R8T8	Read receive data buffer 8 or write transmit data buffer 8.
24 R7T7	Read receive data buffer 7 or write transmit data buffer 7.
25 R6T6	Read receive data buffer 6 or write transmit data buffer 6.
26 R5T5	Read receive data buffer 5 or write transmit data buffer 5.
27 R4T4	Read receive data buffer 4 or write transmit data buffer 4.
28 R3T3	Read receive data buffer 3 or write transmit data buffer 3.
29 R2T2	Read receive data buffer 2 or write transmit data buffer 2.
30 R1T1	Read receive data buffer 1 or write transmit data buffer 1.
31 R0T0	Read receive data buffer 0 or write transmit data buffer 0.

**27.5.5 LPUART Match Address Register (LPUARTx\_MATCH)**

Address: Base address + 10h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Reset 0

**LPUARTx\_MATCH field descriptions**

Field	Description
0–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8–15 MA2	Match Address 2  The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.
16–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24–31 MA1	Match Address 1  The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated BAUD[MAEN] bit is clear.

**27.5.6 LPUART Modem IrDA Register (LPUARTx\_MODIR)**

The MODEM register controls options for setting the modem configuration.

Address: Base address + 14h offset

Bit	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R								0									
W															IREN		TNP
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R								0							RXRTSE		TXRTSPOL
W																TXRTSE	TXCTSE
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

**LPUARTx\_MODIR field descriptions**

Field	Description
0–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 IREN	Infrared enable  Enables/disables the infrared modulation/demodulation.

*Table continues on the next page...*

**LPUARTx\_MODIR field descriptions (continued)**

Field	Description
	<p>0 IR disabled. 1 IR enabled.</p>
14–15 TNP	<p>Transmitter narrow pulse Enables whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse.</p> <p>00 1/OSR. 01 2/OSR. 10 3/OSR. 11 4/OSR.</p>
16–27 Reserved	<p>This field is reserved. This read-only field is reserved and always has the value 0.</p>
28 RXRTSE	<p>Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun.</p> <p><b>NOTE:</b> Do not set both RXRTSE and TXRTSE.</p> <p>0 The receiver has no effect on RTS. 1 RTS assertion is configured by the RTSWATER field</p>
29 TXRTSPOL	<p>Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set.</p> <p>0 Transmitter RTS is active low. 1 Transmitter RTS is active high.</p>
30 TXRTSE	<p>Transmitter request-to-send enable Controls RTS before and after a transmission.</p> <p>0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit.</p>
31 TXCTSE	<p>Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE.</p> <p>0 CTS has no effect on the transmitter. 1 Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.</p>

## 27.5.7 LPUART FIFO Register (LPUARTx\_FIFO)

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the data buffer/FIFO is empty.

Address: Base address + 18h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									TXEMPT	RXEMPT				0	TXOF	RXUF
W															w1c	w1c
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	0									TXFIFOSIZE				RXFIFOSIZE	
W	TXFLUSH	RXFLUSH							TXFE					RXFE		
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0

### LPUARTx\_FIFO field descriptions

Field	Description
0–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 TXEMPT	Transmit Buffer/FIFO Empty  Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register.  0 Transmit buffer is not empty. 1 Transmit buffer is empty.

Table continues on the next page...

**LPUARTx\_FIFO field descriptions (continued)**

Field	Description
9 RXEMPT	<p>Receive Buffer/FIFO Empty</p> <p>Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register.</p> <ul style="list-style-type: none"> <li>0 Receive buffer is not empty.</li> <li>1 Receive buffer is empty.</li> </ul>
10–13 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
14 TXOF	<p>Transmitter Buffer Overflow Flag</p> <p>Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of TXOFE. However, an interrupt will be issued to the host only if TXOFE is set. This flag is cleared by writing a 1.</p> <ul style="list-style-type: none"> <li>0 No transmit buffer overflow has occurred since the last time the flag was cleared.</li> <li>1 At least one transmit buffer overflow has occurred since the last time the flag was cleared.</li> </ul>
15 RXUF	<p>Receiver Buffer Underflow Flag</p> <p>Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of RXUFE. However, an interrupt will be issued to the host only if RXUFE is set. This flag is cleared by writing a 1.</p> <ul style="list-style-type: none"> <li>0 No receive buffer underflow has occurred since the last time the flag was cleared.</li> <li>1 At least one receive buffer underflow has occurred since the last time the flag was cleared.</li> </ul>
16 TXFLUSH	<p>Transmit FIFO/Buffer Flush</p> <p>Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register.</p> <ul style="list-style-type: none"> <li>0 No flush operation occurs.</li> <li>1 All data in the transmit FIFO/Buffer is cleared out.</li> </ul>
17 RXFLUSH	<p>Receive FIFO/Buffer Flush</p> <p>Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register.</p> <ul style="list-style-type: none"> <li>0 No flush operation occurs.</li> <li>1 All data in the receive FIFO/buffer is cleared out.</li> </ul>
18–21 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
22 TXOFE	<p>Transmit FIFO Overflow Interrupt Enable</p> <p>When this field is set, the TXOF flag generates an interrupt to the host.</p> <ul style="list-style-type: none"> <li>0 TXOF flag does not generate an interrupt to the host.</li> <li>1 TXOF flag generates an interrupt to the host.</li> </ul>
23 RXUFE	<p>Receive FIFO Underflow Interrupt Enable</p> <p>When this field is set, the RXUF flag generates an interrupt to the host.</p>

*Table continues on the next page...*

**LPUARTx\_FIFO field descriptions (continued)**

Field	Description
	0 RXUF flag does not generate an interrupt to the host. 1 RXUF flag generates an interrupt to the host.
24 TXFE	Transmit FIFO Enable  When this field is set, the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field.  0 Transmit FIFO is not enabled. Buffer is depth 1. (Legacy support). 1 Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
25–27 TXFIFOSIZE	Transmit FIFO. Buffer Depth  The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only.  101 Transmit FIFO/Buffer depth = 16 datawords.
28 RXFE	Receive FIFO Enable  When this field is set, the built in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field.  0 Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 Receive FIFO is enabled. Buffer is depth indicated by RXFIFOSIZE.
29–31 RXFIFOSIZE	Receive FIFO. Buffer Depth  The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only.  101 Receive FIFO/Buffer depth = 16 datawords.

**27.5.8 LPUART Watermark Register (LPUARTx\_WATER)**

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

Address: Base address + 1Ch offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RXCOUNT							RXWATER							TXCOUNT							TXWATER										
W																																

Reset 0

**LPUARTx\_WATER field descriptions**

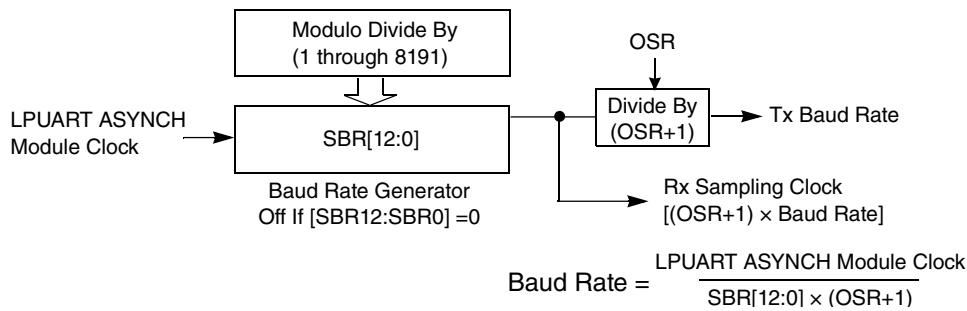
Field	Description
0–7 RXCOUNT	Receive Counter  The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
8–15 RXWATER	Receive Watermark  When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE] and must be greater than 0.
16–23 TXCOUNT	Transmit Counter  The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
24–31 TXWATER	Transmit Watermark  When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

## 27.6 Functional description

The LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the LPUART.

### 27.6.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the baud clock divisor for the asynchronous LPUART baud clock. The SBR bits are in the LPUART baud rate registers, BDH and BDL. The baud rate clock drives the receiver, while the transmitter is driven by the baud rate clock divided by the over sampling ratio. Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.



**Figure 27-3. LPUART baud rate generation**

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

## 27.6.2 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (LPUART\_TX) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the LPUART data register.

The central element of the LPUART transmitter is the transmit shift register that is 10-bit to 13 bits long depending on the setting in the CTRL[M], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at LPUART\_DATA.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the LPUART\_TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with LPUART\_TX high, waiting for more characters to transmit.

Writing 0 to LPUART\_CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

### 27.6.2.1 Send break and queued idle

The LPUART\_CTRL[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting LPUART\_STAT[BRK13]. Normally, a program would wait for LPUART\_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the LPUART\_CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If LPUART\_CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another LPUART, the break characters are received as 0s in all data bits and a framing error (LPUART\_STAT[FE] = 1) occurs.

A break character can also be transmitted by writing to the LPUART\_DATA register with bit 13 set and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows the DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for LPUART\_STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the LPUART\_CTRL[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while LPUART\_CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the LPUART\_TX pin.

An idle character can also be transmitted by writing to the LPUART\_DATA register with bit 13 set and the data bits also set. This supports transmitting the idle character as part of the normal data stream and also allows the DMA to transmit a break character.

The length of the break character is affected by the LPUART\_STAT[BRK13], LPUART\_CTRL[M], LPUART\_BAUD[M10] and LPUART\_BAUD[SNBS] bits as shown below.

**Table 27-5. Break character length**

BRK13	M	M10	SBNS	Break character length
0	0	0	0	10 bit times
0	0	0	1	11 bit times
0	1	0	0	11 bit times
0	1	0	1	12 bit times
0	X	1	0	12 bit times
0	X	1	1	13 bit times
1	0	0	0	13 bit times
1	0	0	1	13 bit times
1	1	0	0	14 bit times
1	1	0	1	14 bit times
1	X	1	0	15 bit times
1	X	1	1	15 bit times

### 27.6.2.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS. If the clear-to-send operation is enabled, the character is transmitted when CTS is asserted. If CTS is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and LPUART\_TX remains in the mark state until CTS is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS.

The transmitter's CTS signal can also be enabled even if the same LPUART receiver's RTS signal is disabled.

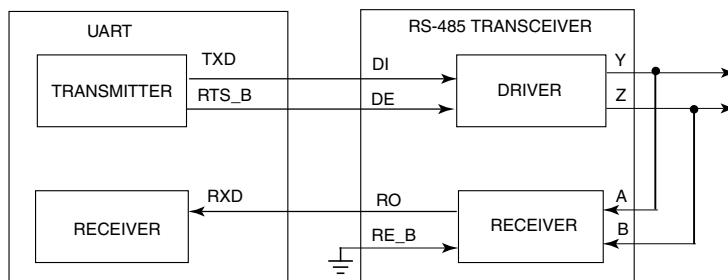
### 27.6.2.3 Transceiver driver enable

The transmitter can use LPUART\_RTS as an enable signal for the driver of an external transceiver. See [Transceiver driver enable using LPUART\\_RTS](#) for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, LPUART\_RTS asserts one bit time before the start bit is transmitted. LPUART\_RTS remains asserted for the whole time that the transmitter data buffer has any characters. LPUART\_RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts LPUART\_RTS, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's LPUART\_RTS signal asserts only when the transmitter is enabled. However, the transmitter's LPUART\_RTS signal is unaffected by its LPUART\_CTS signal. LPUART\_RTS will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

### 27.6.2.4 Transceiver driver enable using LPUART\_RTS

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is 3-stated unless the UART is driving. The LPUART\_RTS signal can be used by the transmitter to enable the driver of a transceiver. The polarity of LPUART\_RTS can be matched to the polarity of the transceiver's driver enable signal.



**Figure 27-4. Transceiver driver enable using LPUART\_RTS**

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect LPUART\_RTS to both DE and RE\_B. The transceiver's receiver is disabled while driving. A pullup can pull LPUART\_RX to a non-floating value during this time. This option can be refined further by operating the LPUART in single wire mode, freeing the LPUART\_RX pin for other uses.

### 27.6.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting LPUART\_STAT[RXINV]. The receiver is enabled by setting the LPUART\_CTRL[RE] bit. Character frames consist of a start bit of logic 0, eight to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 9-bit or 10-bit data mode, refer to [8-bit, 9-bit and 10-bit data modes](#). For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (LPUART\_STAT[RDRF]) status flag is set. If LPUART\_STAT[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the LPUART receiver is double-buffered, the program has one full character time after LPUART\_STAT[RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (LPUART\_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART\_DATA. Refer to [Interrupts and status flags](#) for details about flag clearing.

### 27.6.3.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4 $\times$  and 32 $\times$  of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the LPUART\_RX serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (LPUART\_STAT[NF]) is set when the received character is transferred to the receive data buffer.

When the LPUART receiver is configured to sample on both edges of the baud rate clock, the number of segments in each received bit is effectively doubled (from 1 to OSR $\times$ 2). The start and data bits are then sampled at OSR, OSR+1 and OSR+2. Sampling on both edges of the clock must be enabled for oversampling rates of 4 $\times$  to 7 $\times$  and is optional for higher oversampling rates.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times (unless resynchronization has been disabled). This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

### 27.6.3.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit (LPUART\_CTRL[RWU]). When RWU bit and LPUART\_S2[RWUID] bit are set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force LPUART\_CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver will ignore all characters that do not meet the address match requirements.

**Table 27-6. Receiver Wakeup Options**

RWU	MA1   MA2	WAKE:RWUID	Receiver Wakeup
0	0	X	Normal operation
1	0	00	Receiver wakeup on idle line, IDLE flag not set
1	0	01	Receiver wakeup on idle line, IDLE flag set
1	0	10	Receiver wakeup on address mark
0	1	X0	Address mark address match, IDLE flag not set for discarded characters
0	1	X1	Address mark address match, IDLE flag set for discarded characters

### 27.6.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, LPUART\_CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The LPUART\_CTRL[M] and LPUART\_BAUD[M10] control bit selects 8-bit to 10-bit data mode and the LPUART\_BAUD[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 to 13 bit times because of the start and stop bits.

When LPUART\_CTRL[RWU] is one and LPUART\_STAT[RWUID] is zero, the idle condition that wakes up the receiver does not set the LPUART\_STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the LPUART\_STAT[RDRF] flag and generates an interrupt if enabled. When LPUART\_STAT[RWUID] is one, any idle condition sets the LPUART\_STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether LPUART\_CTRL[RWU] is zero or one.

The idle-line type (LPUART\_CTRL[ILT]) control bit selects one of two ways to detect an idle line. When LPUART\_CTRL[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When LPUART\_CTRL[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

### 27.6.3.2.2 Address-mark wakeup

When LPUART\_CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, LPUART\_CTRL[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character.

Address-mark wakeup allows messages to contain idle characters, but requires the MSB be reserved for use in address frames. The logic 1 in the MSB of an address frame clears the LPUART\_CTRL[RWU] bit before the stop bits are received and sets the LPUART\_STAT[RDRF] flag. In this case, the character with the MSB set is received even though the receiver was sleeping during most of this character time.

### 27.6.3.2.3 Address Match operation

Address match operation is enabled when the LPUART\_BAUD[MAEN1] or LPUART\_BAUD[MAEN2] bit is set and LPUART\_BAUD[MATCFG] is equal to 00. In this function, a character received by the LPUART\_RX pin with a logic 1 in the bit position immediately preceding the stop bit is considered an address and is compared

with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and LPUART\_STAT[RDRF] is set, if the comparison matches. All subsequent characters received with a logic 0 in the bit position immediately preceding the stop bit are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs then no transfer is made to the receive data buffer, and all following characters with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the LPUART\_BAUD[MAEN1] and LPUART\_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of LPUART\_BAUD[MAEN1] and LPUART\_BAUD[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART\_BAUD[MAEN1] and LPUART\_BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

### 27.6.3.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert LPUART\_RTS.

- LPUART\_RTS remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See [Transceiver driver enable using LPUART\\_RTS](#) for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts LPUART\_RTS if the number of characters in the receiver data register is full or a start bit is detected that will cause the receiver data register to be full.
- The receiver asserts LPUART\_RTS when the number of characters in the receiver data register is not full and has not detected a start bit that will cause the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if LPUART\_RTS is deasserted, the receiver continues to receive characters until the receiver data buffer is overrun.
- If the receiver request-to-send functionality is disabled, the receiver LPUART\_RTS remains deasserted.

### 27.6.3.4 Infrared decoder

The infrared decoder converts the received character from the IrDA format to the NRZ format used by the receiver. It also has a OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received.

#### 27.6.3.4.1 Start bit detection

When STAT[RXINV] is cleared, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently from each other.

#### 27.6.3.4.2 Noise filtering

Any further rising edges detected during the first half of the infrared decoder counter are ignored by the decoder. Any pulses less than one oversampling baud clock can be undetected by it regardless of whether it is seen in the first or second half of the count.

#### 27.6.3.4.3 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as a 0, which is sent to the receiver. The decoder counter is also reset.

#### 27.6.3.4.4 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, then the decoder sends a 1 to the receiver.

If the next bit is a 0, which arrives late, then a low-bit is detected according to [Low-bit detection](#). The value sent to the receiver is changed from 1 to a 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, then the delay of a 0 is not recorded as noise.

## 27.6.4 Additional LPUART functions

The following sections describe additional LPUART functions.

### 27.6.4.1 8-bit, 9-bit and 10-bit data modes

The LPUART transmitter and receiver can be configured to operate in 9-bit data mode by setting the LPUART\_CTRL[M] or 10-bit data mode by setting LPUART\_CTRL[M10]. In 9-bit mode, there is a ninth data bit in 10-bit mode there is a tenth data bit. For the transmit data buffer, these bits are stored in LPUART\_CTRL[T8] and LPUART\_CTRL[T9]. For the receiver, these bits are held in LPUART\_CTRL[R8] and LPUART\_CTRL[R9]. They are also accessible via 16-bit or 32-bit accesses to the LPUART\_DATA register.

For coherent 8-bit writes to the transmit data buffer, write to LPUART\_CTRL[T8] and LPUART\_CTRL[T9] before writing to LPUART\_DATA[7:0]. For 16-bit and 32-bit writes to the LPUART\_DATA register all 10 transmit bits are written to the transmit data buffer at the same time.

If the bit values to be transmitted as the ninth and tenth bit of a new character are the same as for the previous character, it is not necessary to write to LPUART\_CTRL[T8] and LPUART\_CTRL[T9] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in LPUART\_CTRL[T8] and LPUART\_CTRL[T9] is copied at the same time data is transferred from LPUART\_DATA[7:0] to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

### 27.6.4.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark position. The CTRL[ILT] register can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

### 27.6.4.3 Loop mode

When LPUART\_CTRL[LOOPS] is set, the LPUART\_CTRL[RSRC] bit in the same register chooses between loop mode (LPUART\_CTRL[RSRC] = 0) or single-wire mode (LPUART\_CTRL[RSRC] = 1). Loop mode is sometimes used to check software,

independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the LPUART\_RX pin is not used by the LPUART.

#### **27.6.4.4 Single-wire operation**

When LPUART\_CTRL[LOOPS] is set, the RSRC bit in the same register chooses between loop mode (LPUART\_CTRL[RSRC] = 0) or single-wire mode (LPUART\_CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the LPUART\_TX pin (the LPUART\_RX pin is not used).

In single-wire mode, the LPUART\_CTRL[TXDIR] bit controls the direction of serial data on the LPUART\_TX pin. When LPUART\_CTRL[TXDIR] is cleared, the LPUART\_TX pin is an input to the receiver and the transmitter is temporarily disconnected from the LPUART\_TX pin so an external device can send serial data to the receiver. When LPUART\_CTRL[TXDIR] is set, the LPUART\_TX pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

#### **27.6.5 Infrared interface**

The LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the LPUART. The IrDA physical layer specification defines a half-duplex infrared communication link for exchanging data. The full standard includes data rates up to 16 Mbits/s. This design covers data rates only between 2.4 kbytes/s and 115.2 kbytes/s.

The LPUART has an infrared transmit encoder and receive decoder. The LPUART transmits serial bits of data that are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses are detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder, external from the LPUART. The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the LPUART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

### 27.6.5.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the LPUART\_TX signal. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent at the start of the bit with a duration of 1/OSR, 2/OSR, 3/OSR, or 4/OSR of a bit time. A narrow low pulse is transmitted for a zero bit when LPUART\_CTRL[TXINV] is cleared, while a narrow high pulse is transmitted for a zero bit when LPUART\_CTRL[TXINV] is set.

### 27.6.5.2 Infrared receive decoder

The infrared receive block converts data from the LPUART\_RX signal to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow low pulse is expected for a zero bit when LPUART\_STAT[RXINV] is cleared, while a narrow high pulse is expected for a zero bit when LPUART\_STAT[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

## 27.6.6 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty LPUART\_STAT[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to LPUART\_DATA. If the transmit interrupt enable LPUART\_CTRL[TIE]) bit is set, a hardware interrupt is requested when LPUART\_STAT[TDRE] is set. Transmit complete (LPUART\_STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with LPUART\_TX at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (LPUART\_CTRL[TCIE]) bit is set, a hardware interrupt is requested when LPUART\_STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the LPUART\_STAT[TDRE] and LPUART\_STAT[TC] status flags if the corresponding LPUART\_CTRL[TIE] or LPUART\_CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (LPUART\_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART\_DATA. The LPUART\_STAT[RDRF] flag is cleared by reading LPUART\_DATA.

## Functional description

The IDLE status flag includes logic that prevents it from getting set repeatedly when the LPUART\_RX line remains idle for an extended period of time. IDLE is cleared by writing 1 to the LPUART\_STAT[IDLE] flag. After LPUART\_STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set LPUART\_STAT[RDRF].

If the associated error was detected in the received character that caused LPUART\_STAT[RDRF] to be set, the error flags - noise flag (LPUART\_STAT[NF]), framing error (LPUART\_STAT[FE]), and parity error flag (LPUART\_STAT[PF]) - are set at the same time as LPUART\_STAT[RDRF]. These flags are not set in overrun cases.

If LPUART\_STAT[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (LPUART\_STAT[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the LPUART\_RX serial data input pin causes the LPUART\_STAT[RXEDGIF] flag to set. The LPUART\_STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (LPUART\_CTRL[RE] = 1).

# Chapter 28

## PCI Express Interface Controller

### 28.1 The PCI Express controller as implemented on the chip

This section provides details about how the PCI Express controller is integrated into this chip.

The following table describes the PCI Express controller integration into this chip:

**Table 28-1. PCI Express controller integration**

Controller	Base address	Maximum number of lanes
PEX1	340_0000h	4
PEX1_LUT	341_0000h	—
PEX2	350_0000h	2
PEX2_LUT	351_0000h	—
PEX3	360_0000h	2
PEX3_LUT	361_0000h	—

The remainder of this chapter refers to a single PCI Express controller offering up to a x4 link interface. Notes are included to indicate variations for multiple instantiations.

#### NOTE

The LS1043A chip is compatible with the PCI Express Base Specification, Revision 3.0; however, the physical layer operates at Gen2 data rates (2.5 or 5 Gbits/s).

#### NOTE

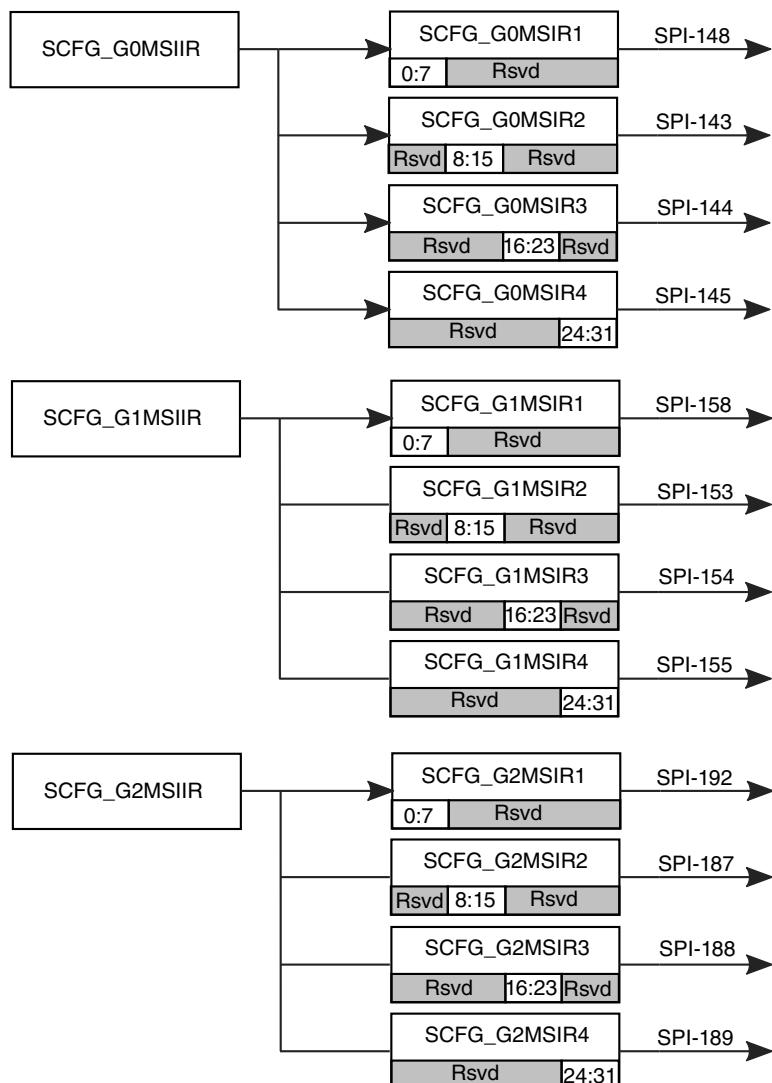
The EP mode is not supported in LS1043A; any references to that should be ignored.

## 28.1.1 PCI Express MSI implementation

The following steps describe the flow sequence of PCI Express MSI implementation:

- Write to SCFG\_GnMSIIR[IBS] sets the index and triggers the corresponding Group MSI interrupt.
- The SCFG\_GnMSIR[SHn] indicates one or more corresponding shared MSI interrupts are pending with the bit numbers selected by the SCFG\_GnMSIIR[IBS] bit field.
- The status register and the MSI interrupt line is cleared on read access to SCFG\_GnMSIR register.

See SCFG\_GnMSIIR and SCFG\_GnMSIR for more information.



**Figure 28-1. PCI Express MSI implementation**

## 28.1.2 PCI Express soft reset support

The PEX\_LUT\_PEXLDBG provides the software configurability to reset PCI Express controllers.

- Write to PEX\_LUT\_PEXLDBG[SR] bit resets the PCI Express 1 controller. This bit needs to be cleared for the reset to be de-asserted.

## 28.1.3 PCI Express PM turnoff message support

The [PEX PME control register \(SCFG\\_PEXPMECR\)](#) provides the software configurability for PM turnoff message. Software needs to set and clear SCFG\_PEXPMECR[PEXnPME] to generate PM turnoff message for power management support for PCI Express controllers.

## 28.1.4 Additional PCI Express events connected to GIC-400 interrupt

The PCI Express controller sideband signals which signify these events are connected to Interrupt lines of GIC-400 interrupt controller. There are no status registers available inside PCI Express controller to indicate these events and the interrupt service routine does not need access to the PCI Express controller CCSR space to clear these events. These Interrupts are to be configured as edge-interrupts.

Internal interrupt ID	Interrupt Source	Comments
235	PEX1 link-down	This is edge triggered interrupt. It detects link down condition. For example, assertion of soft reset triggers this interrupt.
236	PEX1 link-up	This is edge triggered interrupt.
239	PEX2 link-down	This is edge triggered interrupt. It detects link down condition. For example, assertion of soft reset triggers this interrupt.
240	PEX2 link-up	This is edge triggered interrupt. This interrupt is set when link is trained to L0 state. Memory and config accesses should happen after this interrupt is set.
243	PEX3 link-down	This is edge triggered interrupt.

*Table continues on the next page...*

Internal interrupt ID	Interrupt Source	Comments
		It detects link down condition. For example, assertion of soft reset triggers this interrupt.
244	PEX3 link-up	This is edge triggered interrupt. This interrupt is set when link is trained to L0 state. Memory and config accesses should happen after this interrupt is set.

## 28.2 Introduction

The PCI Express interface is compatible with the *PCI Express™ Base Specification, Revision 3.0* (available from <http://www.pcisig.com>). It is beyond the scope of this manual to document the intricacies of the PCI Express protocol. This chapter describes the PCI Express controller of this device and provides a basic description of the PCI Express protocol. The specific emphasis is directed at how the device implements the PCI Express specification. Designers of systems incorporating PCI Express devices should refer to the specification for a thorough description of PCI Express.

### NOTE

Much of the available PCI Express literature refers to a 16-bit quantity as a WORD and a 32-bit quantity as a DWORD. Note that this is inconsistent with the terminology in the rest of this manual where the terms 'word' and 'double word' refer to a 32-bit and 64-bit quantity, respectively. Where necessary to avoid confusion, the precise number of bits or bytes is specified.

### 28.2.1 Overview

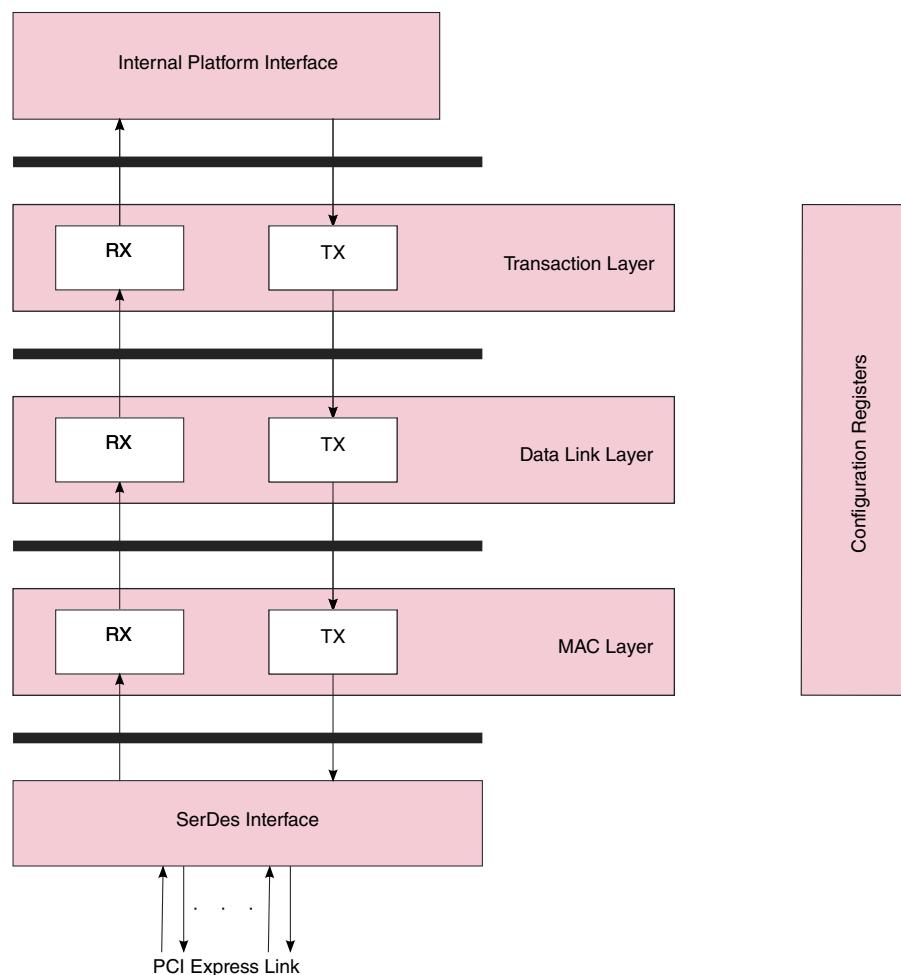
The PCI Express controller connects the internal platform to a serial interface.

As both an initiator and a target device, the PCI Express interface is capable of high-bandwidth data transfer and is designed to support next generation I/O devices. Upon coming out of reset, the PCI Express interface performs link width negotiation and exchanges flow control credits with its link partner. Once link autonegotiation is successful, the controller is in operation.

Internally, the design contains queues to keep track of inbound and outbound transactions. There is control logic that handles buffer management, bus protocol, transaction spawning and tag generation. In addition, there are memory blocks used to store inbound and outbound data.

The PCI Express controller operates as a PCI Express Root Complex (RC) device. An RC device connects the host CPU/memory subsystem to I/O devices. In RC mode, the PCI Express type 1 configuration header is used.

This figure shows a high-level block diagram of the PCI Express controller.



**Figure 28-2. PCI Express Controller Block Diagram**

As an initiator, the PCI Express controller supports memory read and write operations. In addition, configuration and I/O transactions are supported. As a target interface, the PCI Express controller accepts read and write operations to local memory space. Message generation and acceptance are supported. Locked transactions and inbound I/O transactions are not supported.

### 28.2.1.1 Outbound Transactions

Outbound internal platform transactions to PCI Express are first mapped to a translation window to determine what PCI Express transactions are to be issued. A transaction from the internal platform can become a PCI Express Memory, I/O, or Configuration transaction depending on the window attributes.

A transaction may be broken up into smaller sized transactions depending on the original request size, transaction type, and either the PCI Express Device Control register [MAX\_PAYLOAD\_SIZE] field for write requests or the PCI Express Device Control register [MAX\_READ\_SIZE] field for read requests. The controller performs PCI Express ordering rule checking to determine which transaction is to be sent on the PCI Express link.

In general, transactions are serviced in the order that they are received from the internal platform . Only when there is a stalled condition does the controller apply PCI Express ordering rules to outstanding transactions. For posted write transactions, once all data has been received from the internal platform , the data is forwarded to the PCI Express link and the transaction is considered as done. For non-posted write transactions, the controller waits for the completion packets to return before considering the transaction finished. For non-posted read transactions, the controller waits for all completion packets to return and then forwards all data back to the internal platform before terminating the transaction.

Note that after reset or when recovering from a link down condition, external transactions should not be attempted until the link has successfully trained. Software can poll the LTSSM state status to check the status of link training before issuing external requests.

In EP mode, after reset or when recovering from a link down condition, the SoC must not generate any outbound memory or I/O transactions until the remote host has configured the Bus Master Enable bit in the PCI Command register.

### 28.2.1.2 Inbound Transactions

Inbound PCI Express transactions to internal platform are first mapped to a translation window to determine what internal platform transactions are to be issued.

A transaction may be broken up into smaller sized transactions when sending to the internal platform depending on the original request size, byte enables and starting/ending addresses. The controller performs PCI Express ordering rule checking to determine what transaction is to be sent next to the internal platform.

In general, transactions are serviced in the order that they are received from the PCI Express link. Only when there is a stalled condition does the controller apply PCI Express ordering to outstanding transactions. For posted write transactions, once all data has been received from the PCI Express link, the data is forwarded to the internal platform and the transaction is considered as done. For non-posted read transactions, the controller forwards internal platform data back to the PCI Express link.

Note that the controller splits transactions at the crossing of every 256-byte-aligned boundary when sending data back to the PCI Express link.

## 28.2.2 Features

The following is a list of features supported by the PCI Express controller:

- Compatible with the *PCI Express™ Base Specification, Revision 3.0*
- Supports Root Complex (RC) mode
- 32- and 64-bit PCI Express address support
- 40-bit internal platform address support
- x4, x2, and x1 link support.
- Supports accesses to all PCI Express memory and I/O address spaces (requestor only)
- Supports posting of processor-to-PCI Express and PCI Express-to-memory writes
- Supports strong and relaxed transaction ordering rules
- Enforces outbound PCI Express ordering rules and inbound internal platform priority
- PCI Express configuration registers (type 1)
- Baseline and advanced error reporting support
- One virtual channel (VC0)
- 256-byte maximum payload size (MAX\_PAYLOAD\_SIZE)
- Supports 64-bit MSI interrupts. Note that MSI support is provided in the SCFG module.
- Credit-based flow control management handled by PCI Express core.
- Supports PCI Express messages and interrupts
- Accepts up to 256-byte transactions from the internal platform
- Supports Expansion ROM.

## 28.2.3 Modes of Operation

Several parameters that affect the PCI Express controller modes of operation are determined at power-on reset (POR) by reset configuration word (RCW) fields configured depending on SoC product.

**Table 28-2. POR Parameters for PCI Express Controller**

RCW Parameter	Description
SerDes Protocol Select SRDS_PRTCL	Determines the link width
SerDes frequency divider for PCI Express	Determines the link speed

### 28.2.3.1 Link Width

The initial link width is determined by the SerDes protocol configuration field (RCW[SRDS\_PRTCL\_Sn]). See Reset Configuration Word (RCW) for more information. The specific configurations are detailed in SerDes Lane Assignments and Multiplexing.

### 28.2.3.2 Link Speed

The initial link speed is determined by the SerDes frequency divider for PCI Express field (RCW[SRDS\_DIV\_PEX\_Sn]).

See RCW Field Definitions for more information. The specific configurations are detailed in Reference Clocks for SerDes Protocols.

## 28.3 External Signal Descriptions

The PCI Express specification defines the connection between two devices as a link, which can be composed of a single or multiple lanes. Each lane consists of a differential pair for transmitting (TX[n]\_P and TX[n]\_N) and a differential pair for receiving (RX[n]\_P and RX[n]\_N) with an embedded data clock.

**Table 28-3. PCI Express Interface Signals—Detailed Signal Descriptions**

Signal	I/O	Description	
SD_RX[n]_P	I	Receive data, positive. The receive data signals carry PCI Express packet information.	
		<b>State Meaning</b>	Asserted/Negated—Represents data being received from the PCI Express interface.
		<b>Timing</b>	Assertion/Negation—As described in the <i>PCI Express Base Specification, Revision 3.0</i> .
SD_RX[n]_N	I	Receive data, negative. The receive data signals carry PCI Express packet information.	
		<b>State Meaning</b>	Asserted/Negated—Represents the inverse of data being received from the PCI Express interface.

Table continues on the next page...

**Table 28-3. PCI Express Interface Signals—Detailed Signal Descriptions (continued)**

Signal	I/O	Description	
		<b>Timing</b>	Assertion/Negation—As described in the <i>PCI Express Base Specification, Revision 3.0</i> .
SD_TX[n]_P	O	Transmit data, positive. The transmit data signals carry PCI Express packet information.	
		<b>State Meaning</b>	Asserted/Negated—Represents data being transmitted to the PCI Express interface.
		<b>Timing</b>	Assertion/Negation—As described in the <i>PCI Express Base Specification, Revision 3.0</i> .
SD_TX[n]_N	O	Transmit data, negative. The transmit data signals carry PCI Express packet information.	
		<b>State Meaning</b>	Asserted/Negated—Represents the inverse of data being transmitted to the PCI Express interface.
		<b>Timing</b>	Assertion/Negation—As described in the <i>PCI Express Base Specification, Revision 3.0</i> .

## 28.4 Memory map/register overview

### 28.4.1 PCI Express configuration registers

The PCI Express module supports the same configuration registers in both the internal memory map and in PCI Express configuration space. They differ only in whether they are accessed from an internal initiator or from an external initiator on the PCI Express interface. With the exception of the registers in the PEX module internal configuration space, the configuration registers are specified by the PCI Express specification for every PCI Express device. The registers in the PEX module internal configuration space are used for chip-specific functionality and are not defined by the PCI Express specification.

**Table 28-4. PCI Express memory map**

Register space	Offset	NEXT pointer	Notes
Type 1 configuration header	0x000	0x40	RC use Type 1
Power management capability structure	0x040	0x50	
PCI Express capability structure	0x070	0x000 (NULL)	
Advanced error reporting capability structure	0x100	0x148	
Secondary PCI Express capability structure	0x148	0x000 (NULL)	
PEX module internal configuration space	0x700	—	
BAR Mask Registers	0x1000	—	

## 28.4.2 PEX register descriptions

### 28.4.2.1 PCI\_Express\_Configuration\_Registers memory map

PEX1 base address: 340\_0000h

PEX2 base address: 350\_0000h

PEX3 base address: 360\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCI Express Vendor ID Register (Vendor_ID_Register)	16	RO	1957h
2h	PCI Express Device ID Register (Device_ID_Register)	16	RO	8080h
4h	PCI Express Command Register (Command_Register)	16	RW	0000h
6h	PCI Express Status Register (Status_Register)	16	W1C	0010h
8h	PCI Express Revision ID Register (Revision_ID_Register)	8	RO	11h
9h	PCI Express Class Code Register (programming interface) (Class_Code_Register_a)	8	RO	00h
Ah	PCI Express Class Code Register (sub class) (Class_Code_Register_b)	8	RO	20h
Bh	PCI Express Class Code Register (base class) (Class_Code_Register_c)	8	RO	0Bh
Ch	PCI Express Cache Line Size Register (Cache_Line_Size_Register)	8	RW	00h
Dh	PCI Express Latency Timer Register (Latency_Timer_Register)	8	RO	00h
Eh	PCI Express Header Type Register (Header_Type_Register)	8	RO	01h
10h	PCI Express Base Address Register 0 (BAR0)	32	RW	0000_0000h
18h	PCI Express Primary Bus Number Register (Primary_Bus_Number_Register)	8	RW	00h
19h	PCI Express Secondary Bus Number Register (Secondary_Bus_Number_Register)	8	RW	00h
1Ah	PCI Express Subordinate Bus Number Register (Subordinate_Bus_Number_Register)	8	RW	00h
1Ch	PCI Express I/O Base Register (IO_Base_Register)	8	RW	01h
1Dh	PCI Express I/O Limit Register (IO_Limit_Register)	8	RW	01h
1Eh	PCI Express Secondary Status Register (Secondary_Status_Register)	16	W1C	0000h
20h	PCI Express Memory Base Register (Memory_Base_Register)	16	RW	0000h
22h	PCI Express Memory Limit Register (Memory_Limit_Register)	16	RW	0000h
24h	PCI Express Prefetchable Memory Base Register (Prefetchable_Memory_Base_Register)	16	RW	0001h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
26h	PCI Express Prefetchable Memory Limit Register (Prefetchable_Memory_Limit_Register)	16	RW	0001h
28h	PCI Express Prefetchable Base Upper 32 Bits Register (Prefetchable_Base_Upper_32_Bits_Register)	32	RW	0000_0000h
2Ch	PCI Express Prefetchable Limit Upper 32 Bits Register (Prefetchable_Limit_Upper_32_Bits_Register)	32	RW	0000_0000h
30h	PCI Express I/O Base Upper 16 Bits Register (IO_Base_Upper_16_Bits_Register)	16	RO	0000h
32h	PCI Express I/O Limit Upper 16 Bits Register (IO_Limit_Upper_16_Bits_Register)	16	RO	0000h
34h	Capabilities Pointer Register (Capabilities_Pointer_Register)	8	RO	40h
38h	PCI Express Expansion ROM Base Address Register (RC-Mode) (Expansion_ROM_BAR_Type1)	32	RW	0000_0000h
3Ch	PCI Express Interrupt Line Register (Interrupt_Line_Register)	8	RW	FFh
3Dh	PCI Express Interrupt Pin Register (Interrupt_Pin_Register)	8	RO	01h
3Eh	PCI Express Bridge Control Register (Bridge_Control_Register)	16	RW	0000h
40h	PCI Express Power Management Capability ID Register (Power_Management_Capability_ID_Register)	8	RO	01h
42h	PCI Express Power Management Capabilities Register (Power_Management_Capabilities_Register)	16	RO	7E23h
44h	PCI Express Power Management Status and Control Register (Power_Management_Status_and_Control_Register)	16	RW	0000h
47h	PCI Express Power Management Data Register (Power_Management_Data_Register)	8	RO	00h
70h	PCI Express Capability ID Register (Capability_ID_Register)	8	RO	10h
72h	PCI Express Capabilities Register (Capabilities_Register)	16	RO	0042h
74h	PCI Express Device Capabilities Register (Device_Capabilities_Register)	32	RO	0000_8001h
78h	PCI Express Device Control Register (Device_Control_Register)	16	RW	2810h
7Ah	PCI Express Device Status Register (Device_Status_Register)	16	W1C	0000h
7Ch	PCI Express Link Capabilities Register (Link_Capabilities_Register)	32	RO	0073_F443h
80h	PCI Express Link Control Register (Link_Control_Register)	16	RW	0008h
82h	PCI Express Link Status Register (Link_Status_Register)	16	W1C	1000h
84h	PCI Express Slot Capabilities Register (Slot_Capabilities_Register)	32	RO	0000_0000h
88h	PCI Express Slot Control Register (Slot_Control_Register)	16	RW	03C0h
8Ah	PCI Express Slot Status Register (Slot_Status_Register)	16	W1C	0008h
8Ch	PCI Express Root Control Register (Root_Control_Register)	16	RW	0000h
8Eh	PCI Express Root Capabilities Register (Root_Capabilities_Register)	16	RW	0000h
90h	PCI Express Root Status Register (Root_Status_Register)	32	RW	0000_0000h
94h	PCI Express Device Capabilities 2 Register (Device_Capabilities_2_Register)	32	RO	0000_001Fh
98h	PCI Express Device Control 2 Register (Device_Control_2_Register)	16	RW	0000h

Table continues on the next page...

## Memory map/register overview

Offset	Register	Width (In bits)	Access	Reset value
9Ch	PCI Express Link Capabilities 2 Register (Link_Capabilities_2_Register)	32	RO	0000_000Eh
A0h	PCI Express Link Control 2 Register (Link_Control_2_Register)	16	RW	0003h
A2h	PCI Express Link Status 2 Register (Link_Status_2_Register)	16	RO	0000h
100h	PCI Express Advanced Error Reporting Capability ID Register (Advanced_Error_Reportin g_Capability_ID_Register)	16	RO	0001h
104h	PCI Express Uncorrectable Error Status Register (Uncorrectable_Error_Status_Register)	32	W1C	0000_0000h
108h	PCI Express Uncorrectable Error Mask Register (Uncorrectable_Error_Mask_Register)	32	RW	0000_0000h
10Ch	PCI Express Uncorrectable Error Severity Register (Uncorrectable_Error_Severity_Register)	32	RW	0046_2030h
110h	PCI Express Correctable Error Status Register (Correctable_Error_Status_Register)	32	W1C	0000_0000h
114h	PCI Express Correctable Error Mask Register (Correctable_Error_Mask_Register)	32	RW	0000_2000h
118h	PCI Express Advanced Error Capabilities and Control Register (Advanced_Error_Capabilities_and_Control_Register)	32	RW	0000_00A0h
11Ch	PCI Express Header Log Register 1 (Header_Log_Register_DWORD_D1)	32	RO	0000_0000h
120h	PCI Express Header Log Register 2 (Header_Log_Register_DWORD_D2)	32	RO	0000_0000h
124h	PCI Express Header Log Register 3 (Header_Log_Register_DWORD_D3)	32	RO	0000_0000h
128h	PCI Express Header Log Register 4 (Header_Log_Register_DWORD_D4)	32	RO	0000_0000h
12Ch	PCI Express Root Error Command Register (Root_Error_Command_Register)	32	RW	0000_0000h
130h	PCI Express Root Error Status Register (Root_Error_Status_Register)	32	W1C	0000_0000h
134h	PCI Express Correctable Error Source ID Register (Correctable_Error_Source_ID_Register)	16	RO	0000h
136h	PCI Express Error Source ID Register (Error_Source_ID_Register)	16	RO	0000h
148h	Secondary PCI Express Extended Capability Header (SPCIE_CAP_HEADER_REG)	32	RO	0001_0019h
14Ch	Link Control 3 Register (LINK_CONTROL3_REG)	32	RW	0000_0000h
150h	Lane Error Status Register (LANE_ERR_STATUS_REG)	32	W1C	0000_0000h
154h - 15Ah	Lane Equalization Control Register (LANE0_EQUALIZATION_CONTROL - LANE3_EQUALIZATION_CONTROL)	16	RO	See description
71Ch	Symbol Timer Register and Filter Mask 1 Register (SYMBOL_TIMER_FILTER_1_OFF)	32	RW	0000_0280h
8BCh	DBI Read-only Write Enable Register (MISC_CONTROL_1_OFF)	32	RW	0000_0000h
8E0h	Coherency Control Register 1 (COHERENCY_CONTROL_1_OFF)	32	RW	0000_0000h
8E4h	Coherency Control Register 2 (COHERENCY_CONTROL_2_OFF)	32	RW	0000_0000h
8E8h	Coherency Control Register 3 (COHERENCY_CONTROL_3_OFF)	32	RU	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
900h	iATU Index Register (IATU_VIEWPORT_OFF)	32	RW	0000_0000h
904h	iATU Region Control 1 Register (IATU_REGION_CTRL_1_OFF_I_NBOUND_0)	32	RW	0000_0000h
904h	iATU Region Control 1 Register (IATU_REGION_CTRL_1_OFF_O_UTBOUND_0)	32	RW	0000_0000h
908h	iATU Region Control 2 Register (IATU_REGION_CTRL_2_OFF_I_NBOUND_0)	32	RW	0000_0000h
908h	iATU Region Control 2 Register (IATU_REGION_CTRL_2_OFF_O_UTBOUND_0)	32	RW	0000_0000h
90Ch	iATU Lower Base Address Register (IATU_LWR_BASE_ADDR_O_FF_INBOUND_0)	32	RW	0000_0000h
90Ch	iATU Lower Base Address Register (IATU_LWR_BASE_ADDR_O_FF_OUTBOUND_0)	32	RW	0000_0000h
910h	iATU Upper Base Address Register (IATU_UPPER_BASE_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
910h	iATU Upper Base Address Register (IATU_UPPER_BASE_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
914h	iATU Limit Address Register (IATU_LIMIT_ADDR_OFF_INBOUND_0)	32	RW	0000_0FFFh
914h	iATU Limit Address Register (IATU_LIMIT_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0FFFh
918h	iATU Region#N Lower Offset Address Register (IATU_LWR_TAR_GET_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
918h	iATU Outbound Region#N Lower Offset Address Register (IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
91Ch	iATU Upper Target Address Register (IATU_UPPER_TARGET_ADDR_OFF_INBOUND_0)	32	RW	0000_0000h
91Ch	iATU Upper Target Address Register (IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_0)	32	RW	0000_0000h
1010h	Base Address Register 0 Mask (BAR0_MASK)	32	WO	00FF_FFFFh
1014h	Base Address Register 1 Mask (BAR1_MASK)	32	WO	03FF_FFFFh
1038h	Expansion ROM Base Address Register Mask (RC mode) (EXP_ROM_BAR_MASK_RC)	32	WO	00FF_FFFFh

## 28.4.2.2 PCI Express Vendor ID Register (Vendor\_ID\_Register)

### 28.4.2.2.1 Offset

Register	Offset
Vendor_ID_Register	0h

### 28.4.2.2.2 Function

The vendor ID register is used to identify the manufacturer of the device.

#### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

### 28.4.2.2.3 Diagram

Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Vendor_ID																
W																	
Reset	0	0	0	1	1	0	0	1		0	1	0	1	0	1	1	0

### 28.4.2.2.4 Fields

Field	Function
15-0	Vendor ID
Vendor_ID	0x1957 (NXP)

## 28.4.2.3 PCI Express Device ID Register (Device\_ID\_Register)

### 28.4.2.3.1 Offset

Register	Offset
Device_ID_Register	2h

### 28.4.2.3.2 Function

The device ID register is used to identify the device.

#### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

### 28.4.2.3.3 Diagram

Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Device_ID																
W																	
Reset	1	0	0	0	0	0	0	0		1	0	0	0	0	0	0	0

### 28.4.2.3.4 Fields

Field	Function
15-0 Device_ID	1000000010000000b - LS1043A with security 1000000010000001b - LS1043A without security 1000000010001000b - LS1023A with security 1000000010001001b - LS1023A without security

## 28.4.2.4 PCI Express Command Register (Command\_Register)

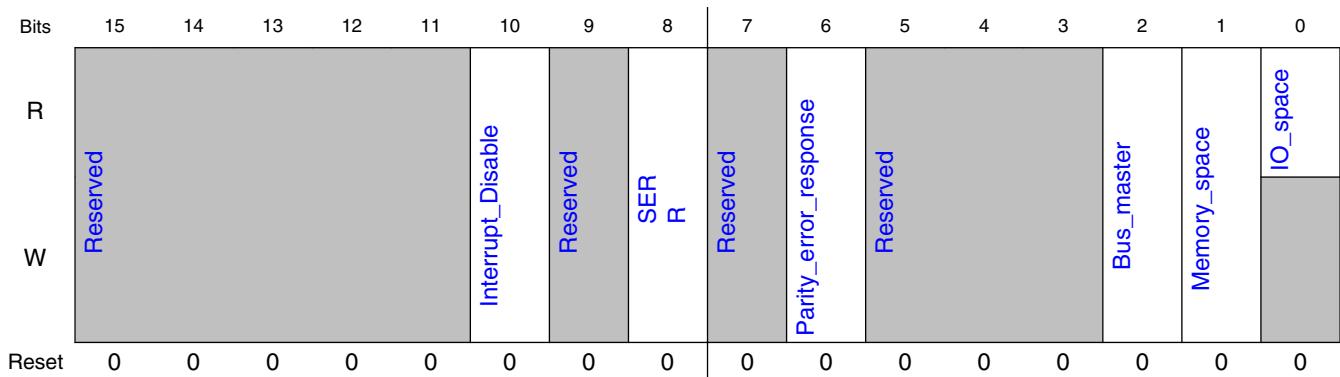
### 28.4.2.4.1 Offset

Register	Offset
Command_Register	4h

### 28.4.2.4.2 Function

The command register provides control over the ability to generate and respond to PCI Express cycles.

### 28.4.2.4.3 Diagram



### 28.4.2.4.4 Fields

Field	Function
15-11 —	Reserved
10 Interrupt_Disable	Interrupt disable Controls the ability to generate INTx interrupt messages. Any INTx emulation interrupts already asserted by this device must be deasserted when this bit is set. 0b - Enables INTx interrupt messages 1b - Disables INTx interrupt messages
9 —	Reserved
8 SERR	SERR# enable Controls the reporting of fatal and non-fatal errors detected by the device to the Root Complex. <b>NOTE:</b> The error control and status bits in the command and status registers control PCI-compatible error reporting. PCI Express advanced error reporting is controlled by the PCI Express device control register described in <a href="#">PCI Express Device Control Register (Device_Control_Register)</a> and the advanced error reporting registers (offsets 100h through 137h). 0b - Disables reporting 1b - Enables reporting
7 —	Reserved
6 Parity_error_response	Parity error response Controls whether this PCI Express controller responds to parity errors. <b>NOTE:</b> The error control and status bits in the command and status registers control PCI-compatible error reporting. PCI Express advanced error reporting is controlled by the PCI Express device control register described in <a href="#">PCI Express Device Control Register (Device_Control_Register)</a> and the advanced error reporting registers (offsets 100h through 137h). 0b - Parity errors are ignored and normal operation continues. 1b - Parity errors cause the appropriate bit in the PCI Express status register to be set. However, note that errors are reported based on the values set in the PCI Express error enable and detection registers.

Table continues on the next page...

Field	Function
5-3 —	Reserved
2 Bus_master	<p>Bus master enable</p> <p>Indicates whether this PCI Express device is configured as a master.</p> <p>RC mode: Clearing this bit disables the ability of the device to forward memory transactions upstream. This causes any inbound memory transaction to be treated as an unsupported request.</p> <p>0b - Disables the ability to generate PCI Express accesses 1b - Enables this PCI Express controller to behave as a PCI Express bus master</p>
1 Memory_space	<p>Memory space enable</p> <p>Controls whether this PCI Express device (as a target) responds to memory accesses.</p> <p>RC mode: This bit is ignored. It does not affect outbound memory transaction</p> <p>0b - This PCI Express device does not respond to PCI Express memory space accesses. 1b - This PCI Express device responds to PCI Express memory space accesses.</p>
0 IO_space	<p>I/O space enable</p> <p>RC mode: This bit is ignored. It does not affect outbound IO transaction.</p> <p>0b - This PCI Express device (as a target) does not respond to PCI Express I/O space accesses. 1b - This PCI Express device (as a target) does respond to PCI Express I/O space accesses.</p>

## 28.4.2.5 PCI Express Status Register (Status\_Register)

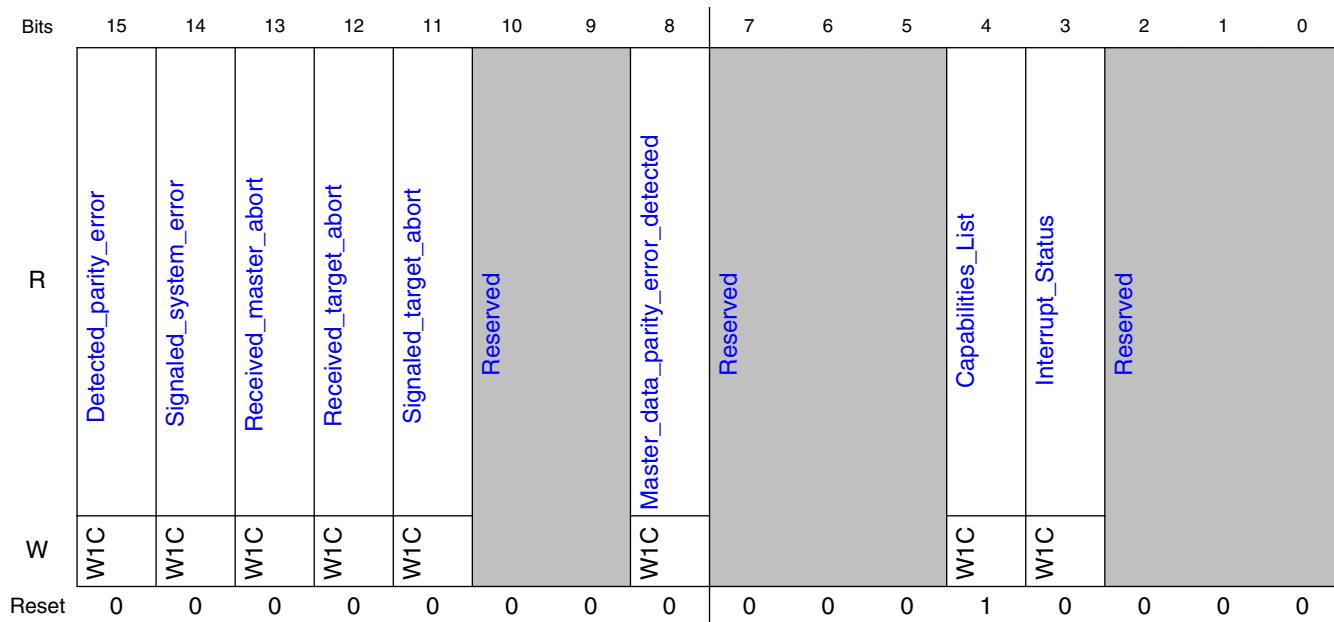
### 28.4.2.5.1 Offset

Register	Offset
Status_Register	6h

### 28.4.2.5.2 Function

The status register is used to record status information for PCI Express related events.

### 28.4.2.5.3 Diagram



### 28.4.2.5.4 Fields

Field	Function
15 Detected_parity_error	Detected parity error Set whenever a device receives a poisoned TLP regardless of the state of bit 6 in the command register. <sup>1</sup>
14 Signaled_system_error	Signaled system error Set whenever a device sends a ERR_FATAL or ERR_NONFATAL message and the SERR enable bit in the command register is set. <sup>1</sup>
13 Received_master_abort	Received master abort Set whenever a requestor receives a completion with unsupported request completion status. <sup>1</sup>
12 Received_target_abort	Received target abort Set whenever a device receives a completion with completer abort completion status. <sup>1</sup>
11 Signaled_target_abort	Signaled target abort Set whenever a device completes a request using completer abort completion status. <sup>1</sup>
10-9 —	Reserved
8 Master_data_parity_error_detected	Master data parity error Set by the requestor (primary side for Type1 headers) when either the requestor receives a completion marked poisoned or the requestor poisons a write request. Note that the parity error enable bit (bit 6) in the command register must be set for this bit to be set. <sup>1</sup>

Table continues on the next page...

Field	Function
7-5 —	Reserved
4 Capabilities_List	Capabilities list All PCI Express devices are required to implement the PCI Express capability structure.
3 Interrupt_Status	Interrupt status Set when an INTx interrupt message is pending internally to the device. Note that this bit is associated with INTx messages and not Message Signaled Interrupts.
2-0 —	Reserved

1. The error control and status bits in the command and status registers control PCI-compatible error reporting. PCI Express advanced error reporting is controlled by the PCI Express device control register described in [PCI Express Device Control Register \(Device\\_Control\\_Register\)](#) and the advanced error reporting capability structure starting at offset 100h.

## 28.4.2.6 PCI Express Revision ID Register (Revision\_ID\_Register)

### 28.4.2.6.1 Offset

Register	Offset
Revision_ID_Register	8h

### 28.4.2.6.2 Function

The revision ID register is used to identify the revision of the device.

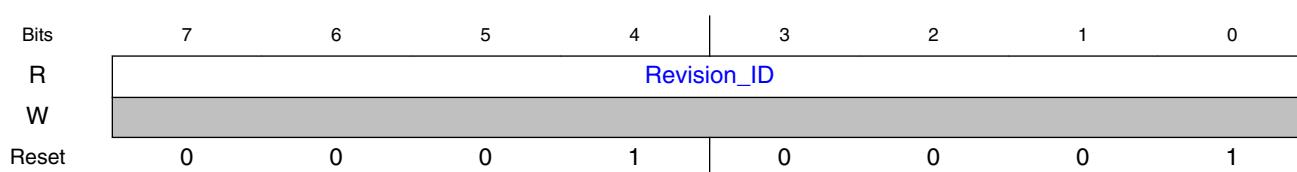
#### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

For Si 1.1, the revision ID is 0x11.

For Si 1.0, the revision ID is 0x10.

### 28.4.2.6.3 Diagram



#### 28.4.2.6.4 Fields

Field	Function
7-0	Revision ID
Revision_ID	Revision specific.

#### 28.4.2.7 PCI Express Class Code Register (programming interface) (Class\_Code\_Register\_a)

##### 28.4.2.7.1 Offset

Register	Offset
Class_Code_Register_a	9h

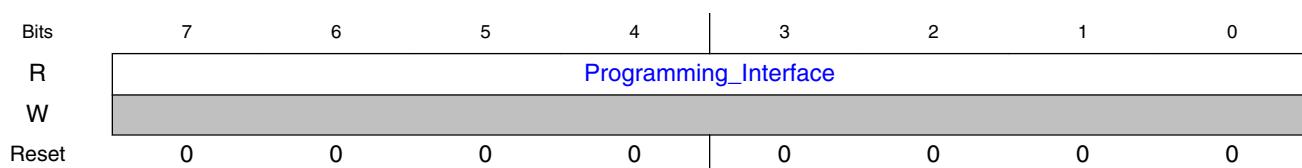
##### 28.4.2.7.2 Function

The class code register is comprised of three single-byte fields—base class (offset 0x0B), sub-class (offset 0x0A), and programming interface (offset 0x09)—that indicate the basic functionality of the function.

##### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

##### 28.4.2.7.3 Diagram



##### 28.4.2.7.4 Fields

Field	Function
7-0	Programming_Interface

Field	Function
Programming_Interface	

## 28.4.2.8 PCI Express Class Code Register (sub class) (Class\_Code\_Register\_b)

### 28.4.2.8.1 Offset

Register	Offset
Class_Code_Register_b	Ah

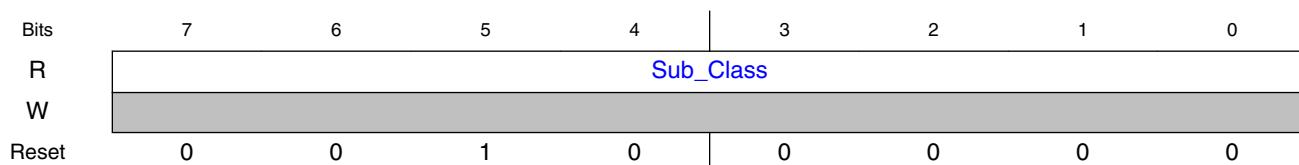
### 28.4.2.8.2 Function

The class code register is comprised of three single-byte fields—base class (offset 0x0B), sub-class (offset 0x0A), and programming interface (offset 0x09)—that indicate the basic functionality of the function.

#### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

### 28.4.2.8.3 Diagram



### 28.4.2.8.4 Fields

Field	Function
7-0 Sub_Class	Sub-Class

## 28.4.2.9 PCI Express Class Code Register (base class) (Class\_Code\_Register\_c)

### 28.4.2.9.1 Offset

Register	Offset
Class_Code_Register_c	Bh

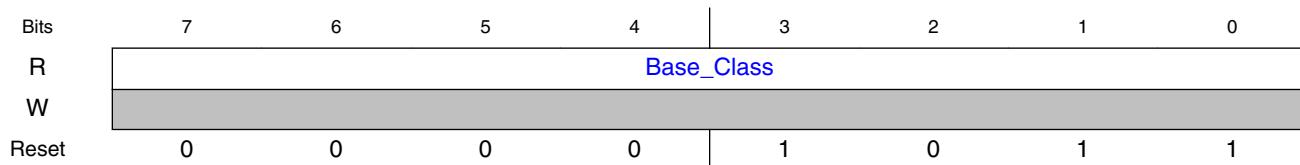
### 28.4.2.9.2 Function

The class code register is comprised of three single-byte fields—base class (offset 0x0B), sub-class (offset 0x0A), and programming interface (offset 0x09)—that indicate the basic functionality of the function.

#### NOTE

This register is writeable using internal accesses, but is read-only from inbound configuration accesses by an external host.

### 28.4.2.9.3 Diagram



### 28.4.2.9.4 Fields

Field	Function
7-0	Base Class
Base_Class	0x0B-Processor

## 28.4.2.10 PCI Express Cache Line Size Register (Cache\_Line\_Size\_Register)

### 28.4.2.10.1 Offset

Register	Offset
Cache_Line_Size_Register	Ch

### 28.4.2.10.2 Function

The cache line size register is provided for legacy compatibility purposes (PCI 2.3); it is not used for PCI Express device functionality.

### 28.4.2.10.3 Diagram



### 28.4.2.10.4 Fields

Field	Function
7-0	Cache Line Size
Cache_Line_Size	Represents the cache line size of the processor in terms of 32-bit words (8 32-bit words = 32 bytes). Note that for PCI Express operation this register is ignored.

## 28.4.2.11 PCI Express Latency Timer Register (Latency\_Timer\_Register)

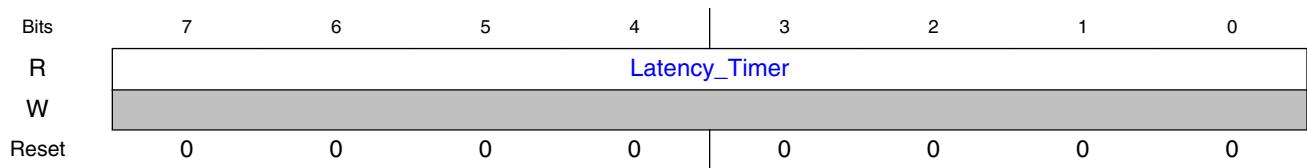
### 28.4.2.11.1 Offset

Register	Offset
Latency_Timer_Register	Dh

### 28.4.2.11.2 Function

The latency timer register is provided for legacy compatibility purposes (PCI 2.3); it is not used for PCI Express device functionality.

### 28.4.2.11.3 Diagram



### 28.4.2.11.4 Fields

Field	Function
7-0	Latency_Timer
Latency_Timer	Note that for PCI Express operation this register is ignored.

## 28.4.2.12 PCI Express Header Type Register (Header\_Type\_Register)

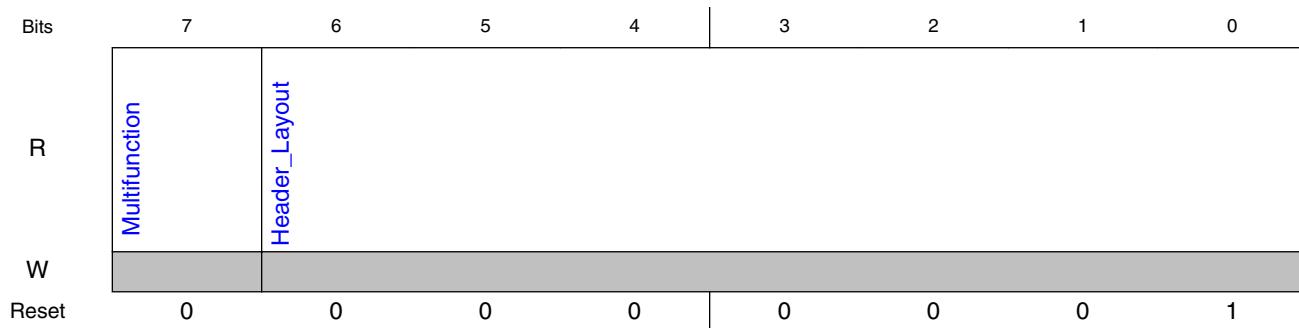
### 28.4.2.12.1 Offset

Register	Offset
Header_Type_Register	Eh

### 28.4.2.12.2 Function

The PCI Express header type register is used to identify the layout of the PCI compatible header.

### 28.4.2.12.3 Diagram



### 28.4.2.12.4 Fields

Field	Function
7 Multifunction	Multifunction Identifies whether a device supports multiple functions 0b - Single function device 1b - Multiple function device
6-0 Header_Layout	Header Layout All other encodings reserved. 0000001b - Root Complex - Type 1 layout.

## 28.4.2.13 PCI Express Base Address Register 0 (BAR0)

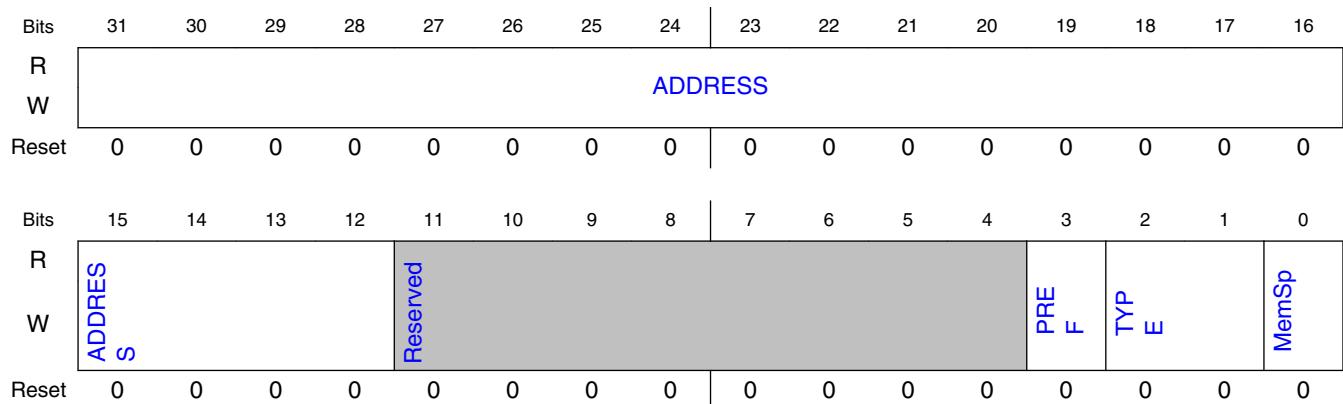
### 28.4.2.13.1 Offset

Register	Offset
BAR0	10h

### 28.4.2.13.2 Function

The PCI Express base address registers (BARs) point to the beginning of distinct address ranges which the device should claim.

### 28.4.2.13.3 Diagram



### 28.4.2.13.4 Fields

Field	Function
31-12 ADDRESS	Base address Indicates the base address of the inbound memory window 0. The default size is 16 MB.
11-4 —	Reserved
3 PREF	Prefetchable
2-1 TYPE	Type 00b - Locate anywhere in 32-bit address space.
0 MemSp	Memory space indicator Base Address registers that map to Memory Space must return a 0 in bit 0. Base Address registers that map to I/O Space must return a 1 in bit 0.

### 28.4.2.14 PCI Express Primary Bus Number Register (Primary\_Bus\_Number\_Register)

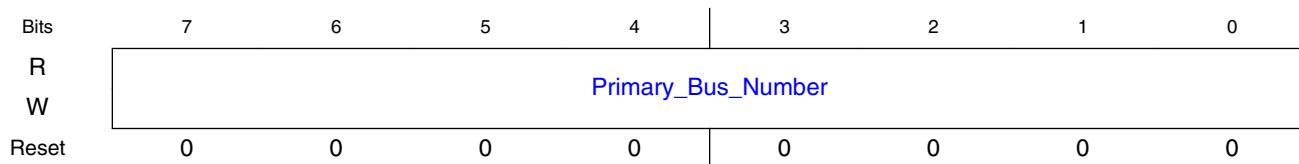
#### 28.4.2.14.1 Offset

Register	Offset
Primary_Bus_Number_Register	18h

### 28.4.2.14.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.14.3 Diagram



### 28.4.2.14.4 Fields

Field	Function
7-0	Primary Bus Number
Primary_Bus_Number	Bus that is connected to the upstream interface. Note that this register is programmed during system enumeration; in RC mode this register should remain 0x00.

## 28.4.2.15 PCI Express Secondary Bus Number Register (Secondary\_Bus\_Number\_Register)

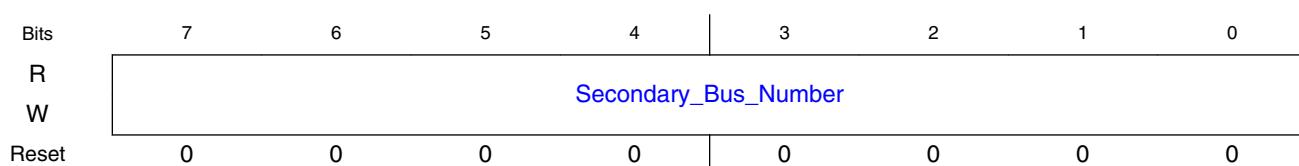
### 28.4.2.15.1 Offset

Register	Offset
Secondary_Bus_Number_Register	19h

### 28.4.2.15.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.15.3 Diagram



### 28.4.2.15.4 Fields

Field	Function
7-0 Secondary_Bus_Number	Secondary Bus Number Bus that is directly connected to the downstream interface. Note that this register is programmed during system enumeration; in RC mode, this register is typically programmed to 0x01.

### 28.4.2.16 PCI Express Subordinate Bus Number Register (Subordinate\_Bus\_Number\_Register)

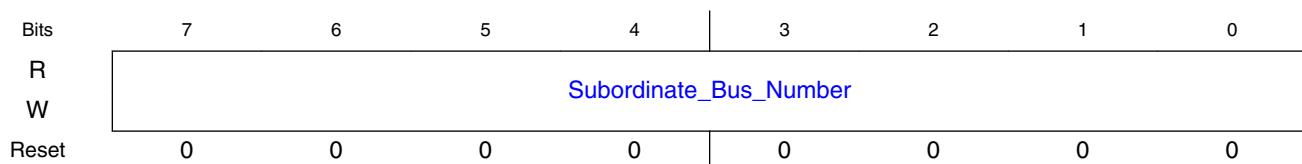
#### 28.4.2.16.1 Offset

Register	Offset
Subordinate_Bus_Number_Register	1Ah

#### 28.4.2.16.2 Function

This register is present only in the Type 1 Header (RC mode).

#### 28.4.2.16.3 Diagram



#### 28.4.2.16.4 Fields

Field	Function
7-0 Subordinate_Bus_Number	Subordinate Bus Number Highest bus number that is on the downstream interface.

## 28.4.2.17 PCI Express I/O Base Register (IO\_Base\_Register)

### 28.4.2.17.1 Offset

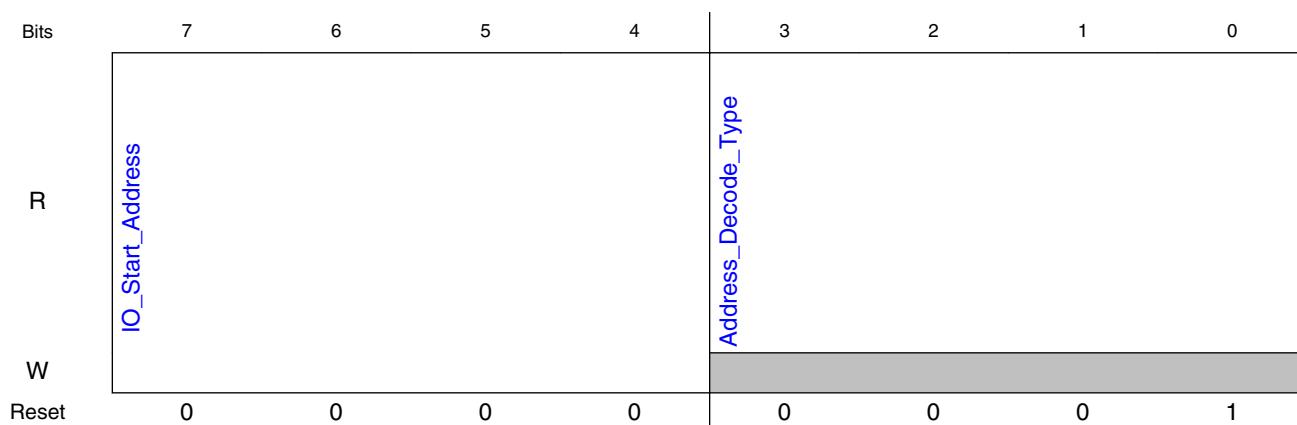
Register	Offset
IO_Base_Register	1Ch

### 28.4.2.17.2 Function

This register is present only in the Type 1 Header (RC mode).

Note that this device does not support inbound I/O transactions.

### 28.4.2.17.3 Diagram



### 28.4.2.17.4 Fields

Field	Function
7-4 IO_Start_Address	I/O Start Address Specifies bits 15:12 of the I/O space start address
3-0 Address_Decode_Type	Address Decode Type Specifies the number of I/O address bits. All other settings are reserved. 0000b - 16-bit I/O address decode 0001b - 32-bit I/O address decode

## 28.4.2.18 PCI Express I/O Limit Register (IO\_Limit\_Register)

### 28.4.2.18.1 Offset

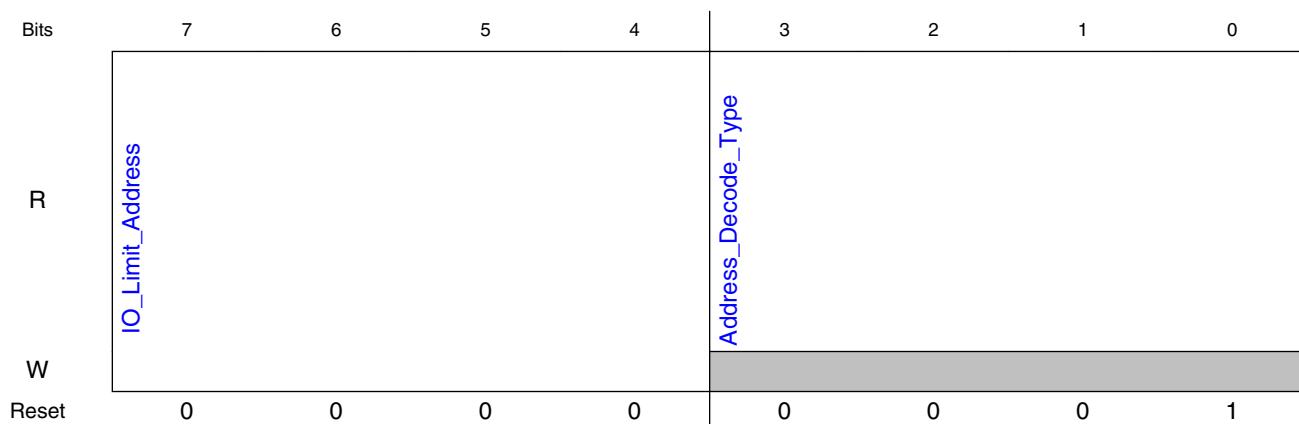
Register	Offset
IO_Limit_Register	1Dh

### 28.4.2.18.2 Function

This register is present only in the Type 1 Header (RC mode).

Note that this device does not support inbound I/O transactions.

### 28.4.2.18.3 Diagram



### 28.4.2.18.4 Fields

Field	Function
7-4 IO_Limit_Address	I/O Limit Address Specifies bits 15:12 of the I/O space ending address
3-0 Address_Decode_Type	Address Decode Type Specifies the number of I/O address bits. All other settings are reserved. 0000b - 16-bit I/O address decode 0001b - 32-bit I/O address decode

## 28.4.2.19 PCI Express Secondary Status Register (Secondary\_Status\_Register)

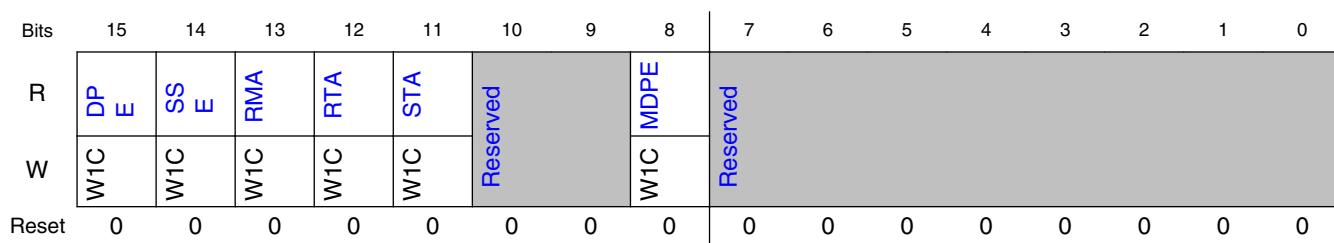
### 28.4.2.19.1 Offset

Register	Offset
Secondary_Status_Register	1Eh

### 28.4.2.19.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.19.3 Diagram



### 28.4.2.19.4 Fields

Field	Function
15 DPE	Detected parity error This bit is set whenever the secondary side receives a poisoned TLP regardless of the state of the parity error response bit.
14 SSE	Signaled system error This bit is set when a device sends a ERR_FATAL or ERR_NONFATAL message, provided the SERR enable bit in the command register is set to enable reporting.
13 RMA	Received master abort This bit is set when the secondary side receives an unsupported request (UR) completion.
12 RTA	Received target abort This bit is set when the secondary side receives a completer abort (CA) completion.
11 STA	Signaled target abort This bit is set when the secondary side issues a CA completion.
10-9	Reserved

Table continues on the next page...

## Memory map/register overview

Field	Function
—	
8	Master data parity error
MDPE	This bit is set when the parity error response bit is set and the secondary side requester receives a poisoned completion or poisons a write request. If the parity error response bit is cleared, this bit is never set.
7-0	Reserved
—	

## 28.4.2.20 PCI Express Memory Base Register (Memory\_Base\_Register)

### 28.4.2.20.1 Offset

Register	Offset
Memory_Base_Register	20h

### 28.4.2.20.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.20.3 Diagram



### 28.4.2.20.4 Fields

Field	Function
15-4 Memory_Base	Memory base address Specifies bits 31:20 of the non-prefetchable memory space start address. Typically used for specifying memory-mapped I/O space.  <b>NOTE:</b> Inbound posted transactions hitting into the mem base/limit range are ignored; inbound non-posted transactions hitting into the mem base/limit range results in an unsupported request response.
3-0	Reserved

Field	Function
—	

## 28.4.2.21 PCI Express Memory Limit Register (Memory\_Limit\_Register)

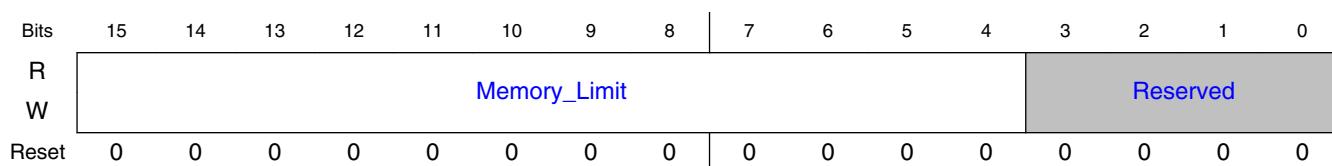
### 28.4.2.21.1 Offset

Register	Offset
Memory_Limit_Register	22h

### 28.4.2.21.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.21.3 Diagram



### 28.4.2.21.4 Fields

Field	Function
15-4 Memory_Limit	Memory limit address Specifies bits 31:20 of the non-prefetchable memory space ending address. Typically used for specifying memory-mapped I/O space.  <b>NOTE:</b> Inbound posted transactions hitting into the mem base/limit range are ignored; inbound non-posted transactions hitting into the mem base/limit range results in unsupported request response.
3-0 —	Reserved

## 28.4.2.22 PCI Express Prefetchable Memory Base Register (Prefetchable\_Memory\_Base\_Register)

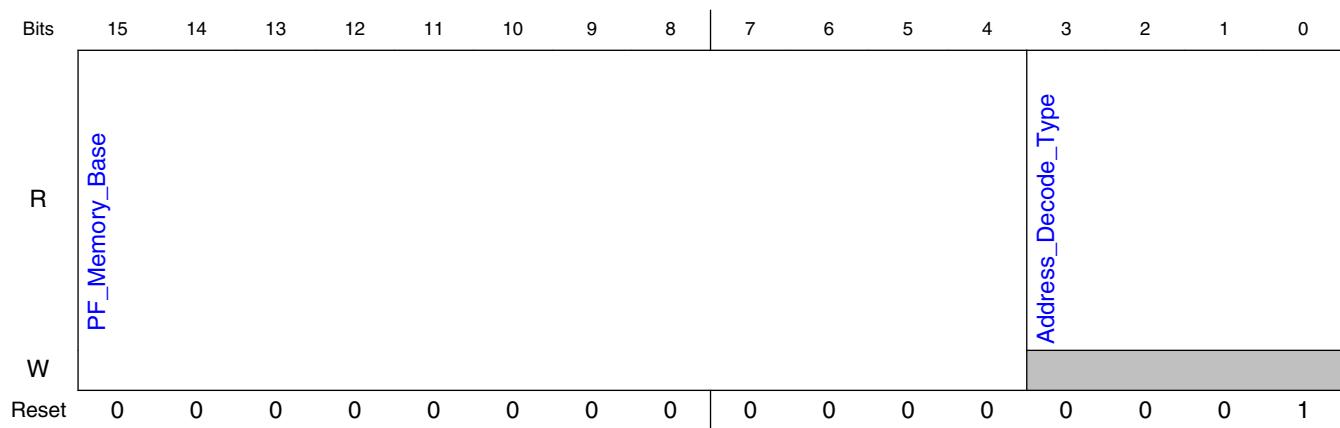
### 28.4.2.22.1 Offset

Register	Offset
Prefetchable_Memory_Base_Register	24h

### 28.4.2.22.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.22.3 Diagram



### 28.4.2.22.4 Fields

Field	Function
15-4 PF_Memory_Base	Prefetchable memory base address Specifies bits 31:20 of the prefetchable memory space start address.
3-0 Address_Decode_Type	Address Decode Type Specifies the number of prefetchable memory address bits. All other settings reserved. 0000b - 32-bit memory address decode 0001b - 64-bit memory address decode

## 28.4.2.23 PCI Express Prefetchable Memory Limit Register (Prefetchable\_Memory\_Limit\_Register)

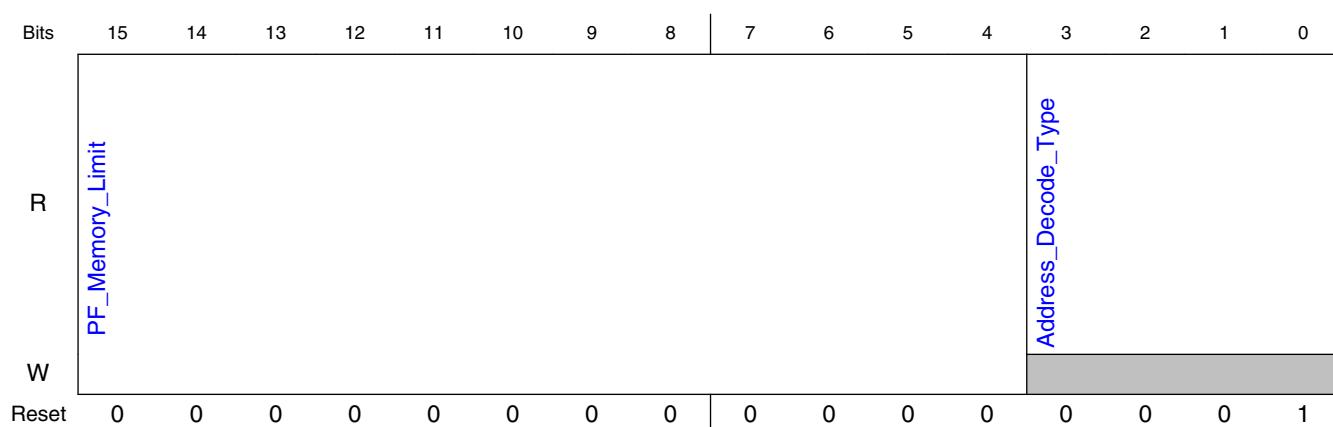
### 28.4.2.23.1 Offset

Register	Offset
Prefetchable_Memory_Limit_Register	26h

### 28.4.2.23.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.23.3 Diagram



### 28.4.2.23.4 Fields

Field	Function
15-4 PF_Memory_Limit	Prefetchable memory limit address Specifies bits 31:20 of the prefetchable memory space ending address.
3-0 Address_Decode_Type	Address decode type Specifies the number of prefetchable memory address bits. All other settings reserved. 0000b - 32-bit memory address decode 0001b - 64-bit memory address decode

## 28.4.2.24 PCI Express Prefetchable Base Upper 32 Bits Register (Prefetchable\_Base\_Upper\_32\_Bits\_Register)

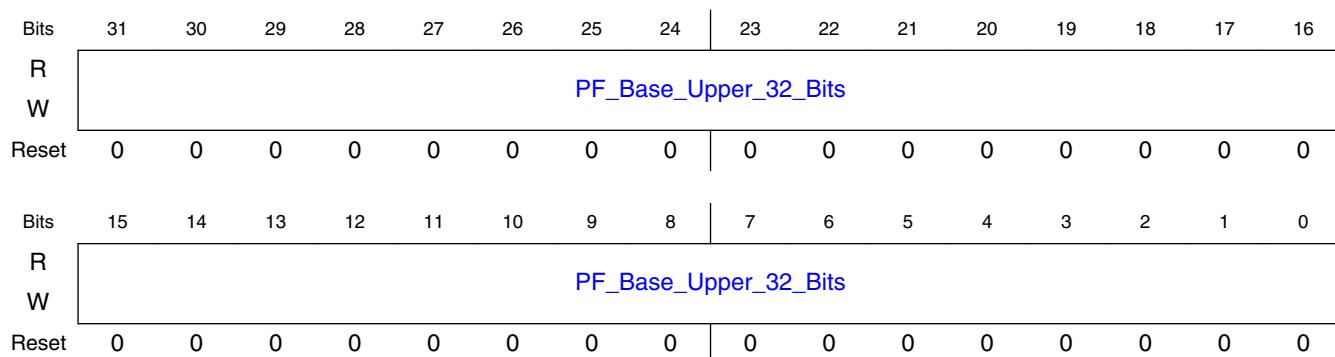
### 28.4.2.24.1 Offset

Register	Offset
Prefetchable_Base_Upper_32_Bits_Register	28h

### 28.4.2.24.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.24.3 Diagram



### 28.4.2.24.4 Fields

Field	Function
31-0	Prefetchable memory base address (upper portion)
PF_Base_Upper_32_Bits	Specifies bits 64:32 of the prefetchable memory space start address when the address decode type field in the prefetchable memory base register is 0x01.

## 28.4.2.25 PCI Express Prefetchable Limit Upper 32 Bits Register (Prefetchable\_Limit\_Upper\_32\_Bits\_Register)

### 28.4.2.25.1 Offset

Register	Offset
Prefetchable_Limit_Upper_32_Bits_Register	2Ch

### 28.4.2.25.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.25.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R																	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 28.4.2.25.4 Fields

Field	Function
31-0	Prefetchable memory limit address (upper portion)
PF_Limit_Upper_32_Bits	Specifies bits 64-32 of the prefetchable memory space ending address when the address decode type field in the prefetchable memory limit register is 0x01.

## 28.4.2.26 PCI Express I/O Base Upper 16 Bits Register (IO\_Base\_Upper\_16\_Bits\_Register)

### 28.4.2.26.1 Offset

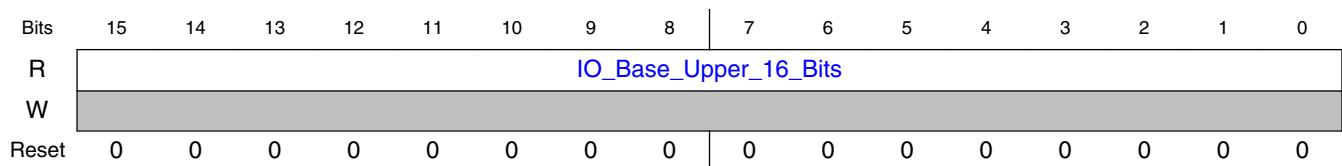
Register	Offset
IO_Base_Upper_16_Bits_Register	30h

### 28.4.2.26.2 Function

This register is present only in the Type 1 Header (RC mode).

Note that this device does not support inbound I/O transactions.

### 28.4.2.26.3 Diagram



### 28.4.2.26.4 Fields

Field	Function
15-0	I/O base address (upper portion)
IO_Base_Upper_16_Bits	Specifies bits 31-16 of the I/O space start address when the address decode type field in the I/O base register is 0x01.

## 28.4.2.27 PCI Express I/O Limit Upper 16 Bits Register (IO\_Limit\_Upper\_16\_Bits\_Register)

### 28.4.2.27.1 Offset

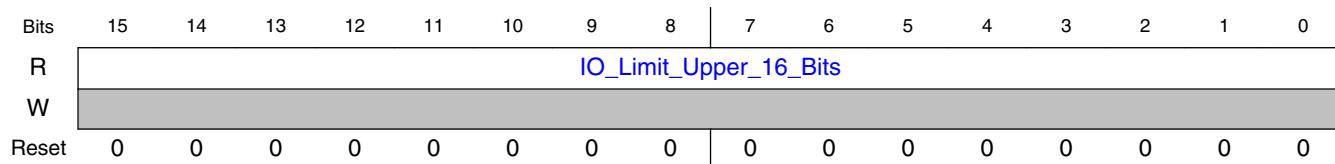
Register	Offset
IO_Limit_Upper_16_Bits_Register	32h

### 28.4.2.27.2 Function

This register is present only in the Type 1 Header (RC mode).

Note that this device does not support inbound I/O transactions.

### 28.4.2.27.3 Diagram



### 28.4.2.27.4 Fields

Field	Function
15-0 IO_Limit_Upper _16_Bits	I/O limit address (upper portion) Specifies bits 31-16 of the I/O space ending address when the address decode type field in the I/O limit register is 0x01.

## 28.4.2.28 Capabilities Pointer Register (Capabilities\_Pointer\_Register)

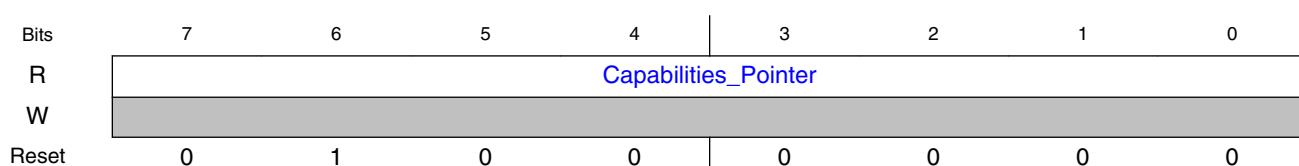
### 28.4.2.28.1 Offset

Register	Offset
Capabilities_Pointer_Reg ister	34h

### 28.4.2.28.2 Function

The capabilities pointer identifies additional functionality supported by the device.

### 28.4.2.28.3 Diagram



## 28.4.2.28.4 Fields

Field	Function
7-0	Capabilities Pointer
Capabilities_Pointer	The capabilities pointer provides the offset for additional PCI-compatible registers above the common 64-byte header.

## 28.4.2.29 PCI Express Expansion ROM Base Address Register (RC-Mode) (Expansion\_ROM\_BAR\_Type1)

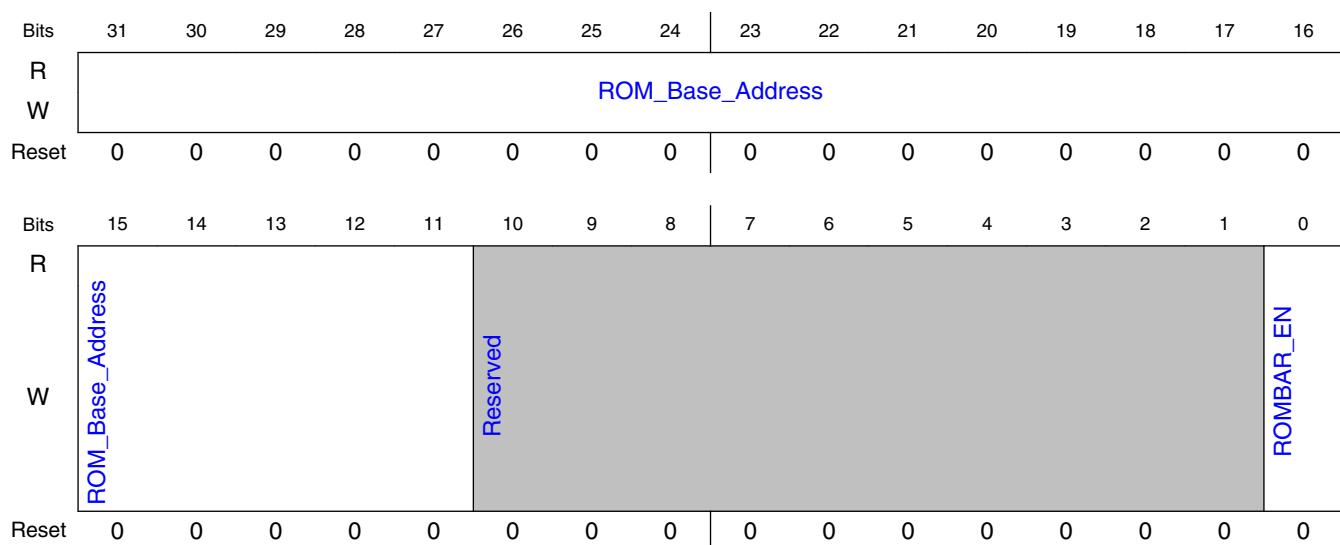
### 28.4.2.29.1 Offset

Register	Offset
Expansion_ROM_BAR_Type1	38h

### 28.4.2.29.2 Function

The Expansion ROM Base Address register is located at offset 0x38 in the Type 1 Header (RC mode).

### 28.4.2.29.3 Diagram



### 28.4.2.29.4 Fields

Field	Function
31-11 ROM_Base_Ad dress	Expansion ROM base address Specifies bits 31:11 of the non-prefetchable expansion ROM space start address. Typically used for specifying memory-mapped I/O space. The default size is 16M.
10-1 —	Reserved
0 ROMBAR_EN	Expansion ROM enable This bit controls whether or not the device accepts accesses to its expansion ROM 0b - The expansion ROM address space is disabled. 1b - Address decoding is enabled.

### 28.4.2.30 PCI Express Interrupt Line Register (Interrupt\_Line\_Regis ter)

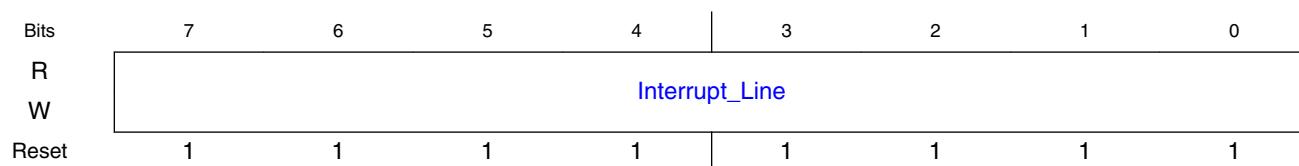
#### 28.4.2.30.1 Offset

Register	Offset
Interrupt_Line_Register	3Ch

#### 28.4.2.30.2 Function

The interrupt line register is used by device drivers and OS software to communicate interrupt line routing information. Values in this register are programmed by system software and are system specific.

#### 28.4.2.30.3 Diagram



## 28.4.2.30.4 Fields

Field	Function
7-0	Interrupt line
Interrupt_Line	Used to communicate interrupt line routing information.

## 28.4.2.31 PCI Express Interrupt Pin Register (Interrupt\_Pin\_Register)

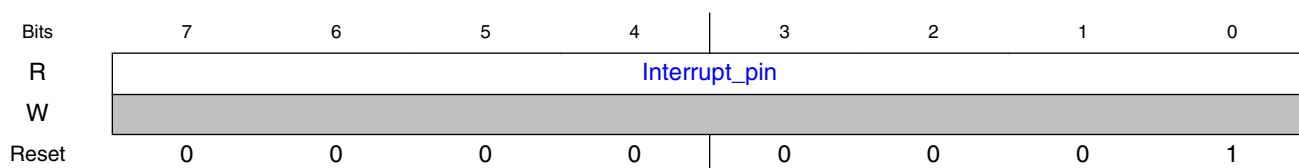
### 28.4.2.31.1 Offset

Register	Offset
Interrupt_Pin_Register	3Dh

### 28.4.2.31.2 Function

The interrupt pin register identifies the legacy interrupt (INTx) messages the device (or function) uses.

### 28.4.2.31.3 Diagram



### 28.4.2.31.4 Fields

Field	Function
7-0 Interrupt_pin	Interrupt pin Legacy INTx message used by this device. All other settings reserved. 00000000b - This device does not use legacy interrupt (INTx) messages. 00000001b - INTA

## 28.4.2.32 PCI Express Bridge Control Register (Bridge\_Control\_Register)

### 28.4.2.32.1 Offset

Register	Offset
Bridge_Control_Register	3Eh

### 28.4.2.32.2 Function

This register is present only in the Type 1 Header (RC mode).

### 28.4.2.32.3 Diagram



### 28.4.2.32.4 Fields

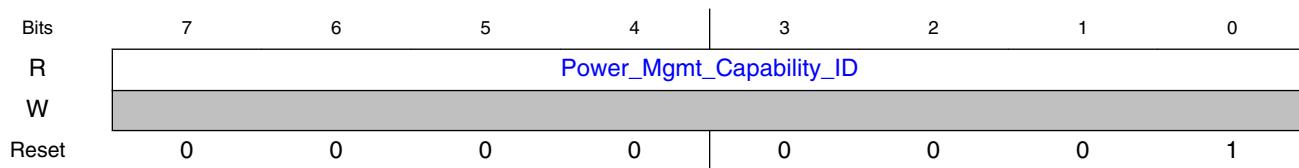
Field	Function
15-7 —	Reserved
6 Scnd_RST	Secondary bus reset
5-4 —	Reserved
3 VGA_EN	VGA enable
2 ISA_EN	ISA enable
1 SERR_EN	SERR enable This bit controls the propagation of ERR_COR, ERR_NONFATAL, and ERR_FATAL responses received on the secondary side.
0 PER	Parity error response This bit controls the logging of poisoned TLPs in the Master Data Parity Error bit in the Secondary Status register.

### 28.4.2.33 PCI Express Power Management Capability ID Register (Power\_Management\_Capability\_ID\_Register)

#### 28.4.2.33.1 Offset

Register	Offset
Power_Management_Capability_ID_Register	40h

#### 28.4.2.33.2 Diagram



#### 28.4.2.33.3 Fields

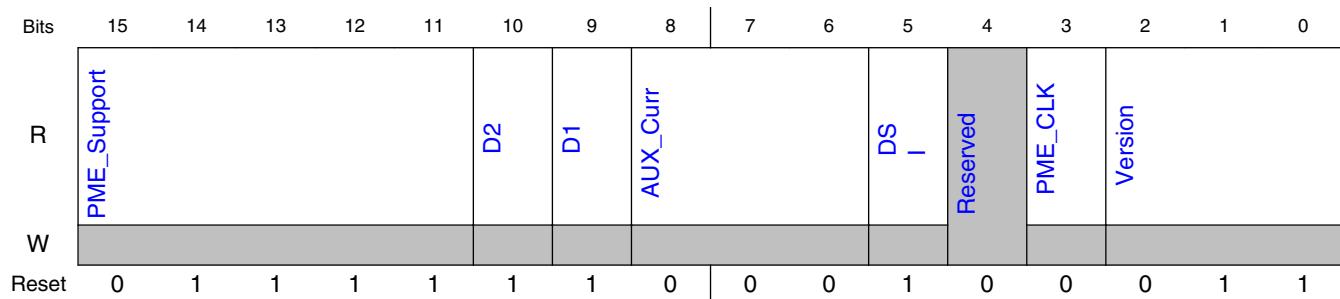
Field	Function
7-0 Power_Mgmt_Capability_ID	CAP_ID Power Management = 0x01

### 28.4.2.34 PCI Express Power Management Capabilities Register (Power\_Management\_Capabilities\_Register)

#### 28.4.2.34.1 Offset

Register	Offset
Power_Management_Capabilities_Register	42h

### 28.4.2.34.2 Diagram



### 28.4.2.34.3 Fields

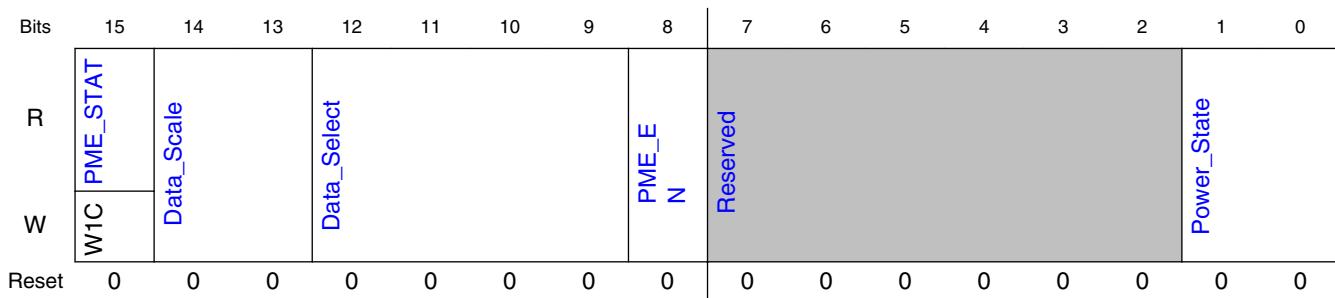
Field	Function
15-11 PME_Support	PME support Indicates the power states that this device supports
10 D2	D2 support
9 D1	D1 support
8-6 AUX_Curr	AUX Current
5 DSI	Device Specific Initialization
4 —	Reserved
3 PME_CLK	PME clock Does not apply to PCI Express.
2-0 Version	Version

### 28.4.2.35 PCI Express Power Management Status and Control Register (Power\_Management\_Status\_and\_Control\_Register)

### 28.4.2.35.1 Offset

Register	Offset
Power_Management_Status_and_Control_Register	44h

### 28.4.2.35.2 Diagram



### 28.4.2.35.3 Fields

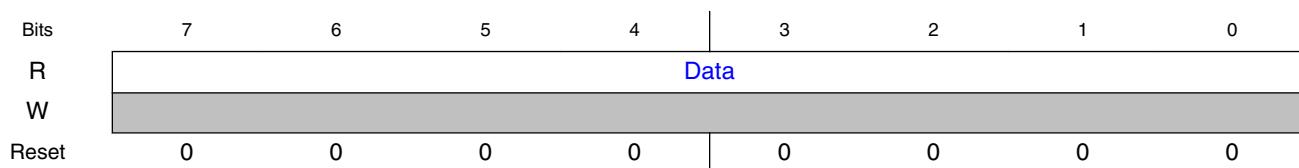
Field	Function
15 PME_STAT	PME Status
14-13 Data_Scale	Data Scale Obtained directly from the PCI Express base specification.
12-9 Data_Select	Data Select Obtained directly from the PCI Express base specification.
8 PME_EN	PME_En PME Enable. <b>NOTE:</b> Bitfield access is sticky.
7-2 —	Reserved
1-0 Power_State	Power State Indicates the current power state of the function. 00b - D0 01b - D1 10b - D2 11b - D3

## 28.4.2.36 PCI Express Power Management Data Register (Power\_Management\_Data\_Register)

### 28.4.2.36.1 Offset

Register	Offset
Power_Management_Data_Register	47h

### 28.4.2.36.2 Diagram



### 28.4.2.36.3 Fields

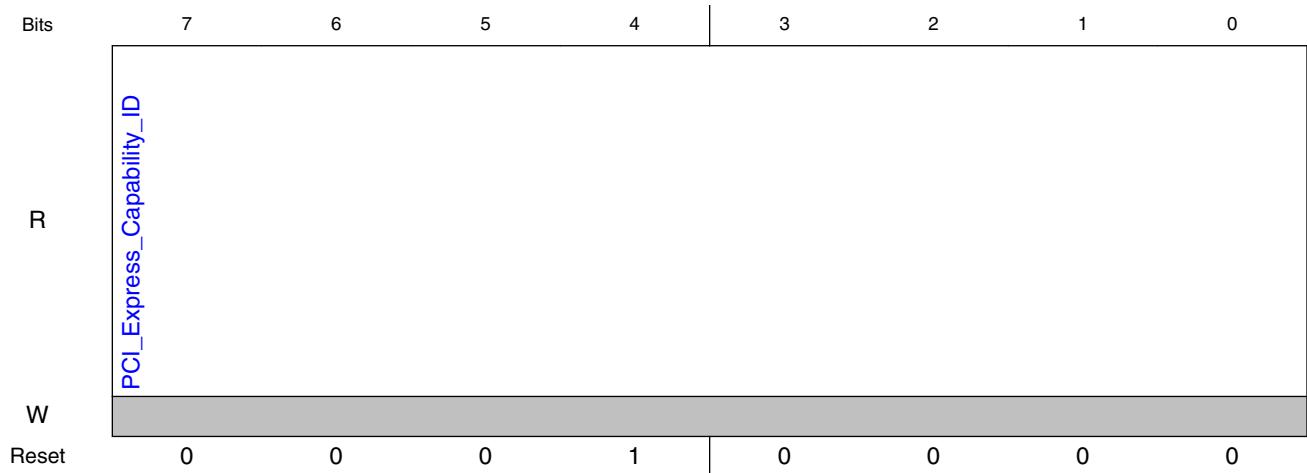
Field	Function
7-0	Data
Data	Obtained from the PCI Express base specification.

## 28.4.2.37 PCI Express Capability ID Register (Capability\_ID\_Register)

### 28.4.2.37.1 Offset

Register	Offset
Capability_ID_Register	70h

## 28.4.2.37.2 Diagram



## 28.4.2.37.3 Fields

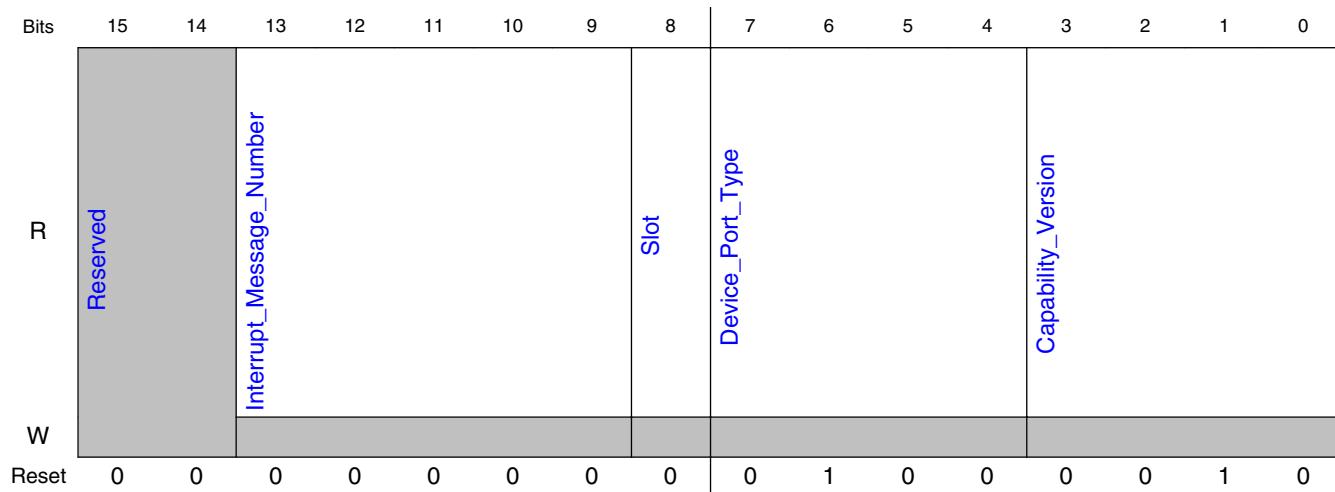
Field	Function
7-0 PCI_Express_Capability_ID	Capability ID PCI Express = 0x10

## 28.4.2.38 PCI Express Capabilities Register (Capabilities\_Register)

### 28.4.2.38.1 Offset

Register	Offset
Capabilities_Register	72h

### 28.4.2.38.2 Diagram



### 28.4.2.38.3 Fields

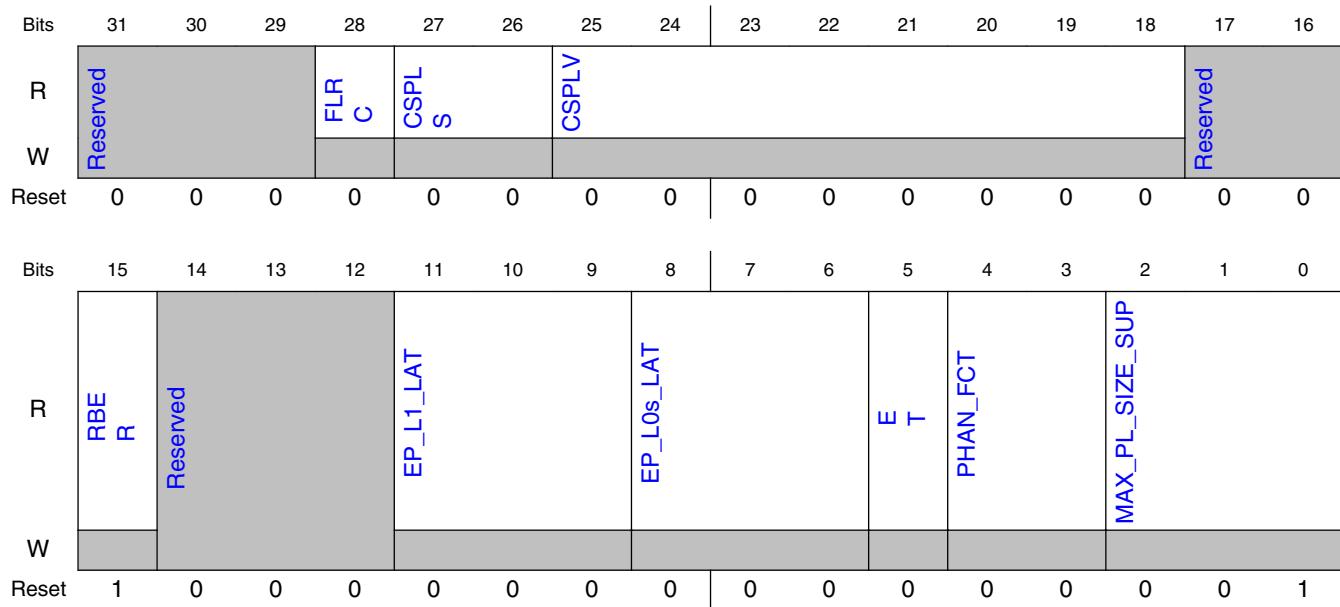
Field	Function
15-14 —	Reserved
13-9 Interrupt_Message_Number	Interrupt Message Number If this function is allocated more than one MSI interrupt number, then this register is required to contain the offset between the base Message Data and the MSI Message that is generated when any of the status bits in either the Slot Status register or the Root Port Status register, of this capability structure, are set.
8 Slot	Slot Implemented (RC mode only)
7-4 Device_Port_Type	Device/Port Type 0100b - (RC mode)
3-0 Capability_Version	Capability Version Indicates the defined PCI Express capability structure version number.

### 28.4.2.39 PCI Express Device Capabilities Register (Device\_Capabilities\_Register)

### 28.4.2.39.1 Offset

Register	Offset
Device_Capabilities_Register	74h

### 28.4.2.39.2 Diagram



### 28.4.2.39.3 Fields

Field	Function
31-29 —	Reserved
28 FLRC	Function Level Reset Capability For RC mode, the reset value for this bit is 0.
27-26 CSPLS	Captured Slot Power Limit Scale
25-18 CSPLV	Captured Slot Power Limit Value For RC mode, the reset value for this field is 00h.
17-16 —	Reserved
15 RBER	Role-Based Error Reporting

Table continues on the next page...

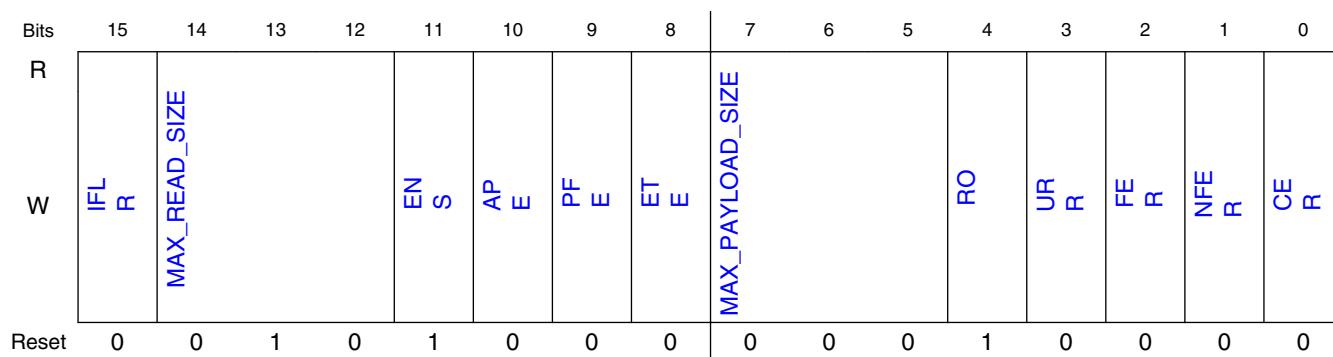
Field	Function
14-12 —	Reserved. System software must ignore the value read from these bits. System software is permitted to write any value to these bits.
11-9 EP_L1_LAT	Endpoint L1 Acceptable Latency
8-6 EP_L0s_LAT	Endpoint L0s Acceptable Latency
5 ET	Extended Tag Field Supported
4-3 PHAN_FCT	Phantom Functions Supported
2-0 MAX_PL_SIZE_SUP	Max_Payload_Size Supported Maximum payload size supported. 001 = 256-bytes

## 28.4.2.40 PCI Express Device Control Register (Device\_Control\_Register)

### 28.4.2.40.1 Offset

Register	Offset
Device_Control_Register	78h

### 28.4.2.40.2 Diagram



### 28.4.2.40.3 Fields

Field	Function
15 IFLR	Initiate Function Level Reset
14-12 MAX_READ_SIZE	Maximum_Read_Request_Size
11 ENS	Enable No Snoop
10 APE	AUX Power PM Enable
9 PFE	Phantom Functions Enable
8 ETE	Extended Tag Field Enable
7-5 MAX_PAYLOAD_SIZE	Max_Payload_Size Maximum payload size
4 RO	Enable Relaxed Ordering
3 URR	Unsupported Request Reporting Enable
2 FER	Fatal Error Reporting Enable
1 NFER	Non-Fatal Error Reporting Enable
0 CER	Correctable Error Reporting Enable

### 28.4.2.41 PCI Express Device Status Register (Device\_Status\_Register)

#### 28.4.2.41.1 Offset

Register	Offset
Device_Status_Register	7Ah

### 28.4.2.41.2 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										TP	APD	URD	FED	NFED	CED
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.41.3 Fields

Field	Function
15-6	Reserved
—	
5 TP	Transactions Pending
4 APD	AUX Power Detected
3 URD	Unsupported Request Detected
2 FED	Fatal Error Detected
1 NFED	Non-Fatal Error Detected
0 CED	Correctable Error Detected

### 28.4.2.42 PCI Express Link Capabilities Register (Link\_Capabilities\_Register)

#### 28.4.2.42.1 Offset

Register	Offset
Link_Capabilities_Register	7Ch

## 28.4.2.42.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Port_Number								Reserved	AOC	LBWN	DLLARC	SD_ERR_RPT_CAP	CPM	L1_EX_LAT	
W									0	1	1	1	0	0	1	1
Reset	0	0	0	0	0	0	0	0	0	1	5	4	3	2	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	L1_EX_LAT	L0s_EX_LAT			ASPM		MAX_LINK_W						MAX_LINK_SP			
W									0	1	0	0	0	0	1	1
Reset	1	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1

## 28.4.2.42.3 Fields

Field	Function
31-24 Port_Number	Port Number This field indicates the PCI Express Port number for the given PCI Express Link
23 —	Reserved
22 AOC	ASPM Optionality Compliance Software is permitted to use the value of this bit to help determine whether to enable ASPM or whether to run ASPM compliance tests.
21 LBWN	Link Bandwidth Notification Capability In RC-mode it is hardwired to 1.
20 DLLARC	Data Link Layer Active Reporting Capable Set to 1 when in RC.
19 SD_ERR_RPT_CAP	Surprise Down Error Reporting Capable
18 CPM	Clock Power Management
17-15 L1_EX_LAT	L1 Exit Latency

Table continues on the next page...

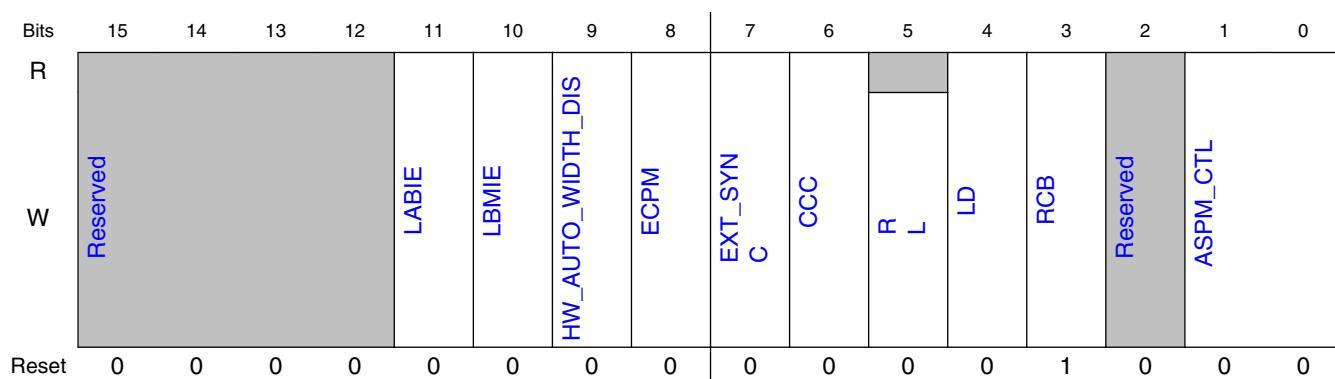
Field	Function
14-12 L0s_EX_LAT	L0s Exit Latency
11-10 ASPM	Active State Power Management (ASPM) Support
9-4 MAX_LINK_W	Maximum Link Width
3-0 MAX_LINK_SP	Maximum Link Speed 0001b - 2.5 GT/s link 0010b - 5.0 GT/s

## 28.4.2.43 PCI Express Link Control Register (Link\_Control\_Register)

### 28.4.2.43.1 Offset

Register	Offset
Link_Control_Register	80h

### 28.4.2.43.2 Diagram



### 28.4.2.43.3 Fields

Field	Function
15-12 —	Reserved
11	Link Autonomous Bandwidth Interrupt Enable

Table continues on the next page...

## Memory map/register overview

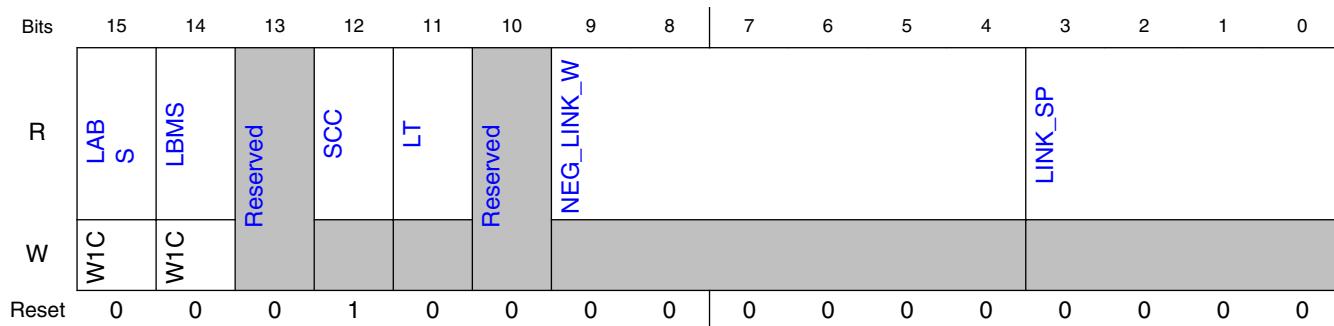
Field	Function
LABIE	
10	Link Bandwidth Management Interrupt Enable
LBMIE	
9	Hardware Autonomous Width Disable
HW_AUTO_WID	
TH_DIS	
8	Enable Clock Power Management
ECPM	
7	Extended Synch
EXT_SYNC	
6	Common Clock Configuration
CCC	
5	Retrain Link
RL	In RC mode, setting this bit initiates link retraining by directing the Physical Layer LTSSM to the Recovery state; reads of this bit always return 0.
4	Link Disable
LD	
3	Read Completion Boundary (RCB)
RCB	
2	Reserved
—	
1-0	Active State Power Management (ASPM) Control
ASPM_CTL	

## 28.4.2.44 PCI Express Link Status Register (Link\_Status\_Register)

### 28.4.2.44.1 Offset

Register	Offset
Link_Status_Register	82h

### 28.4.2.44.2 Diagram



### 28.4.2.44.3 Fields

Field	Function
15 LABS	Link Autonomous Bandwidth Status <b>NOTE:</b> This bit is write-1-clear in RC mode
14 LBMS	Link Bandwidth Management Status <b>NOTE:</b> This bit is write-1-clear in RC mode.
13 —	Reserved
12 SCC	Slot Clock Configuration
11 LT	Link Training
10 —	Reserved.
9-4 NEG_LINK_W	Negotiated link width All other encodings are reserved. The value in this field is undefined when the link is not up. 000001b - x1 000010b - x2 000100b - x4
3-0 LINK_SP	Current Link Speed 0001b - 2.5 GT/s 0010b - 5.0 GT/s

### 28.4.2.45 PCI Express Slot Capabilities Register (Slot\_Capabilities\_Register)

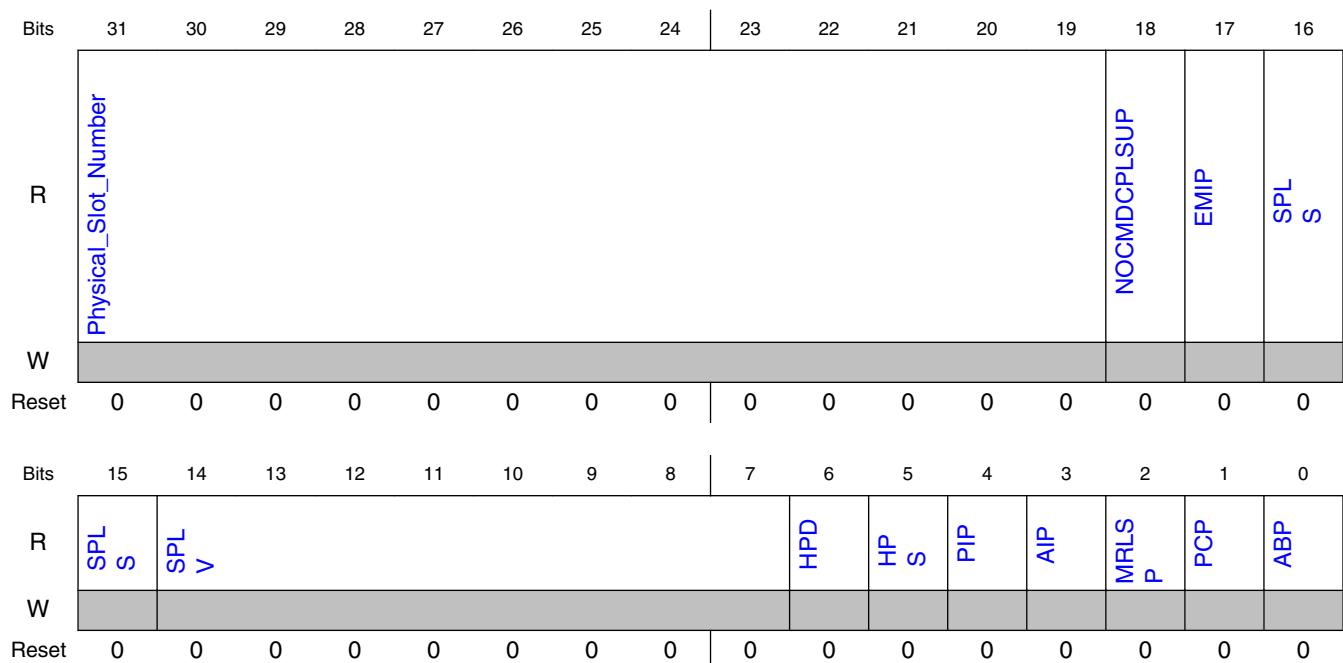
### 28.4.2.45.1 Offset

Register	Offset
Slot_Capabilities_Register	84h

### 28.4.2.45.2 Function

This register is supported only for RC mode.

### 28.4.2.45.3 Diagram



### 28.4.2.45.4 Fields

Field	Function
31-19 Physical_Slot_Number	Physical Slot Number This field indicates the physical slot number attached to this Port. This field must be hardware initialized to a value that assigns a slot number that is globally unique within the chassis. This field must be initialized to 0 for Ports connected to devices that are either integrated on the system board or integrated within the same silicon as the Switch device or Root Port.
18 NOCMDCPLSUP	No Command Completed Support

Table continues on the next page...

Field	Function
17 EMIP	Electromechanical Interlock Present
16-15 SPLS	Slot Power Limit Scale
14-7 SPLV	Slot Power Limit Value
6 HPD	Hot-Plug Capable <b>Note:</b> This chip does not support hot-plug capabilities.
5 HPS	Hot-Plug Surprise <b>Note:</b> This chip does not support hot-plug capabilities.
4 PIP	Power Indicator Present <b>Note:</b> This chip does not support hot-plug capabilities.
3 AIP	Attention Indicator Present <b>Note:</b> This chip does not support hot-plug capabilities.
2 MRLSP	MRL Sensor Present <b>Note:</b> This chip does not support hot-plug capabilities.
1 PCP	Power Controller Present <b>Note:</b> This chip does not support hot-plug capabilities.
0 ABP	Attention Button Present <b>Note:</b> This chip does not support hot-plug capabilities.

## 28.4.2.46 PCI Express Slot Control Register (Slot\_Control\_Register)

### 28.4.2.46.1 Offset

Register	Offset
Slot_Control_Register	88h

### 28.4.2.46.2 Function

This register is supported only for RC mode.

### 28.4.2.46.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
R	Reserved								DLLSTCHGEN	EMICTL	PCC	PIC	AIC	HPI E	CCIE	PDCE	MRLSC E	PFDE	ABP E
W	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	

### 28.4.2.46.4 Fields

Field	Function
15-13 —	Reserved
12 DLLSTCHGEN	Data Link Layer State Changed Enable <b>Note:</b> This chip does not support hot-plug capabilities.
11 EMICTL	Electromechanical Interlock Control <b>Note:</b> This chip does not support hot-plug capabilities.
10 PCC	Power Controller Control <b>Note:</b> This chip does not support hot-plug capabilities.
9-8 PIC	Power Indicator Control <b>Note:</b> This chip does not support hot-plug capabilities.
7-6 AIC	Attention Indicator Control <b>Note:</b> This chip does not support hot-plug capabilities.
5 HPIE	Hot-Plug Interrupt Enable <b>Note:</b> This chip does not support hot-plug capabilities.
4 CCIE	Command Completed Interrupt Enable <b>Note:</b> This chip does not support hot-plug capabilities.
3 PDCE	Presence Detect Changed Enable <b>Note:</b> This chip does not support hot-plug capabilities.
2 MRLSCE	MRL Sensor Changed Enable <b>Note:</b> This chip does not support hot-plug capabilities.
1 PFDE	Power Fault Detected Enable <b>Note:</b> This chip does not support hot-plug capabilities.
0 ABPE	Attention Button Pressed Enable <b>Note:</b> This chip does not support hot-plug capabilities.

## 28.4.2.47 PCI Express Slot Status Register (Slot\_Status\_Register)

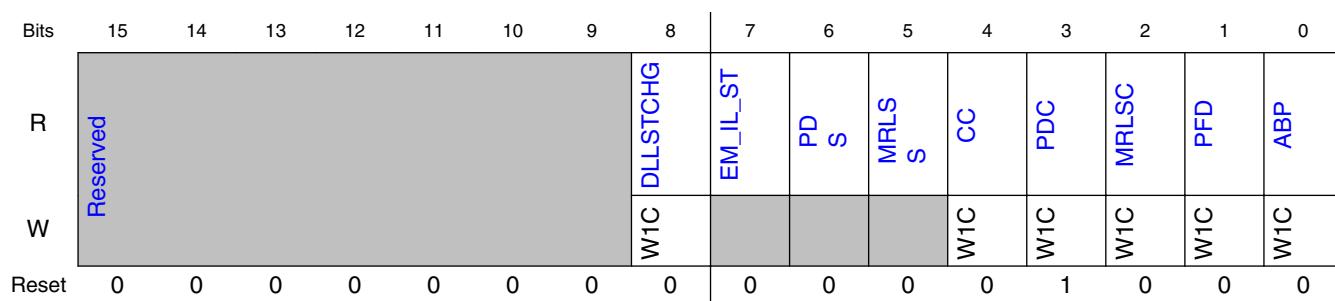
### 28.4.2.47.1 Offset

Register	Offset
Slot_Status_Register	8Ah

### 28.4.2.47.2 Function

This register is supported only for RC mode.

### 28.4.2.47.3 Diagram



### 28.4.2.47.4 Fields

Field	Function
15-9 —	Reserved
8 DLLSTCHG	Data Link Layer State Changed
7 EM_IL_ST	Electromechanical Interlock Status
6 PDS	<p>Presence Detect State</p> <p>This bit indicates the presence of an adapter in the slot, reflected by the logical OR of the Physical Layer in-band presence detect mechanism and, if present, any out-of-band presence detect mechanism defined for the slot's corresponding form factor. Note that the in-band presence detect mechanism requires that power be applied to an adapter for its presence to be detected. Consequently, form factors that require a power controller for hot-plug must implement a physical pin presence detect mechanism.</p> <p>This register must be implemented on all Downstream Ports that implement slots. For Downstream Ports not connected to slots (where the Slot Implemented bit of the PCI Express Capabilities register is 0b), this bit must return 1b.</p> <p>Defined encodings are:</p>

*Table continues on the next page...*

## Memory map/register overview

Field	Function
	0b - Slot Empty 1b - Card Present in slot
5 MRLSS	MRL Sensor State 0b - MRL closed 1b - MRL open
4 CC	Command Completed
3 PDC	Presence Detect Changed
2 MRLSC	MRL Sensor Changed
1 PFD	Power Fault Detected
0 ABP	Attention Button Pressed

## 28.4.2.48 PCI Express Root Control Register (Root\_Control\_Register)

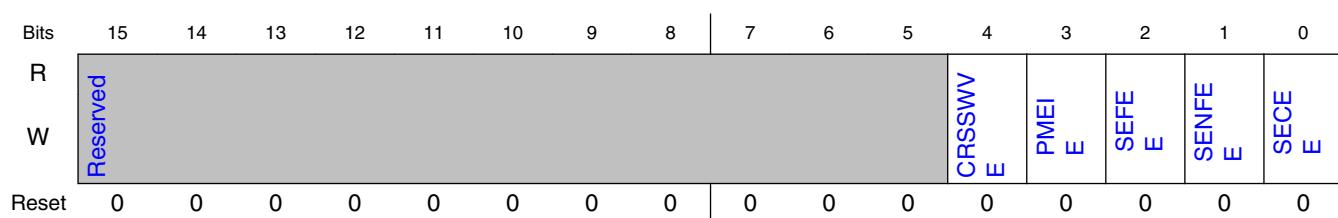
### 28.4.2.48.1 Offset

Register	Offset
Root_Control_Register	8Ch

### 28.4.2.48.2 Function

This register is supported only for RC mode.

### 28.4.2.48.3 Diagram



### 28.4.2.48.4 Fields

Field	Function
15-5 —	Reserved
4 CRSSWVE	CRS Software Visibility Enable
3 PMEIE	PME Interrupt Enable
2 SEFEE	System Error on Fatal Error Enable
1 SENFEE	System Error on Non-Fatal Error Enable
0 SECEE	System Error on Correctable Error Enable

### 28.4.2.49 PCI Express Root Capabilities Register (Root\_Capabilities\_Register)

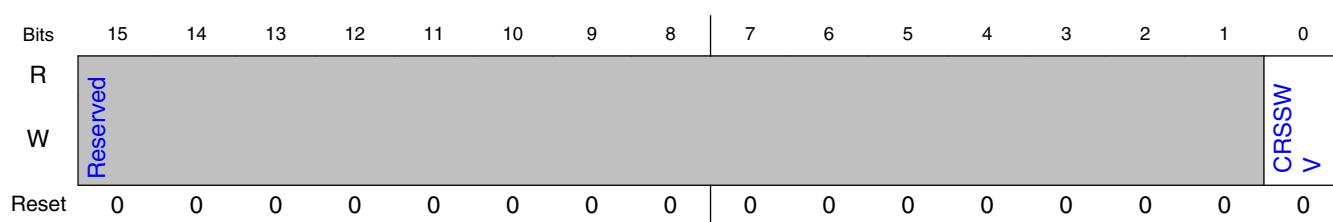
#### 28.4.2.49.1 Offset

Register	Offset
Root_Capabilities_Register	8Eh

#### 28.4.2.49.2 Function

This register is supported only for RC mode.

#### 28.4.2.49.3 Diagram



### 28.4.2.49.4 Fields

Field	Function
15-1 —	Reserved
0 CRSSWV	CRS Software Visibility

### 28.4.2.50 PCI Express Root Status Register (Root\_Status\_Register)

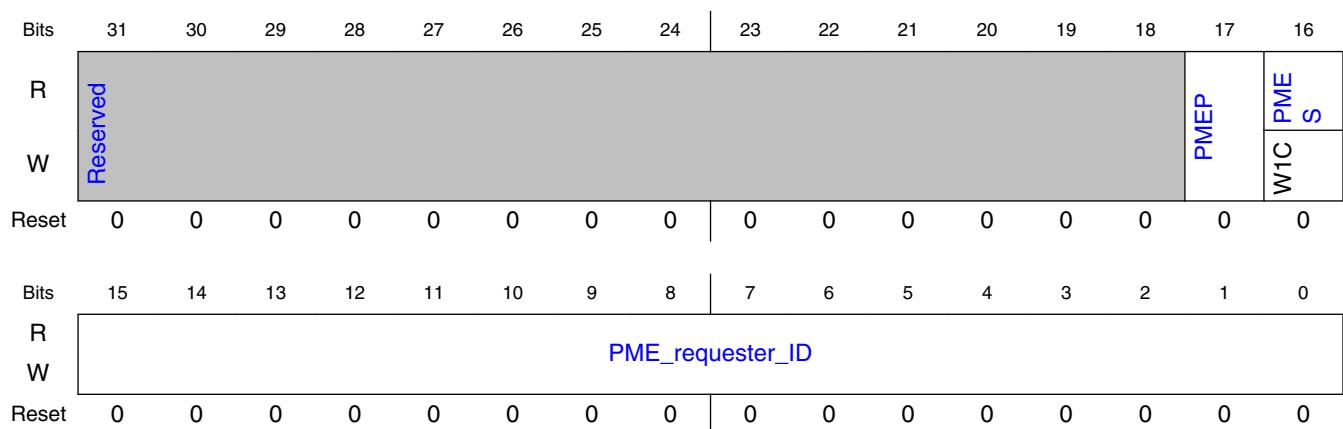
#### 28.4.2.50.1 Offset

Register	Offset
Root_Status_Register	90h

#### 28.4.2.50.2 Function

This register is supported only for RC mode.

#### 28.4.2.50.3 Diagram



#### 28.4.2.50.4 Fields

Field	Function
31-18	Reserved

*Table continues on the next page...*

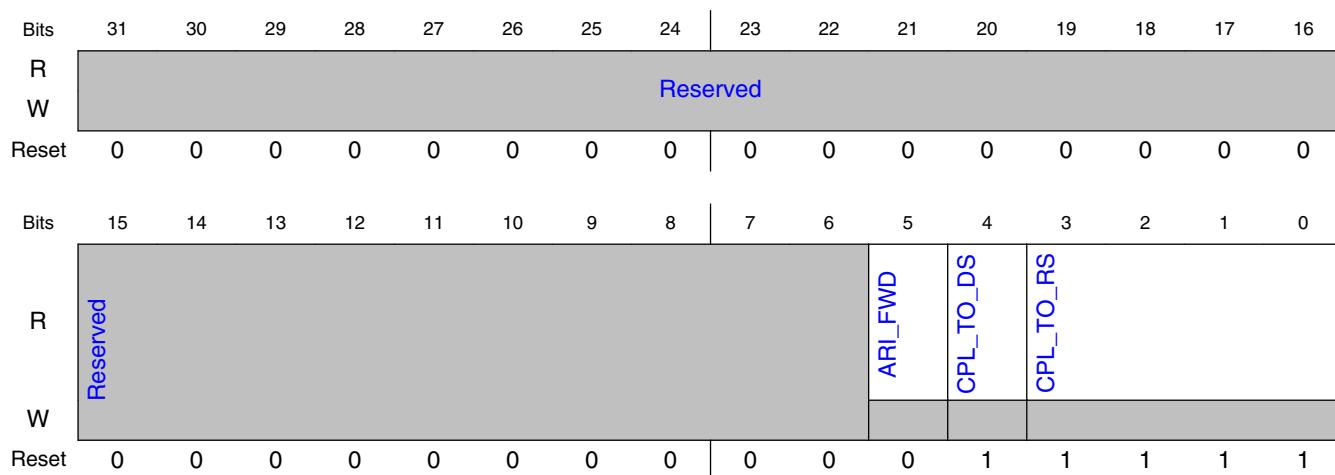
Field	Function
—	
17 PMEP	PME Pending
16 PMES	PME Status
15-0 PME_requester _ID	PME Requester ID

### 28.4.2.51 PCI Express Device Capabilities 2 Register (Device\_Capabilities\_2\_Register)

#### 28.4.2.51.1 Offset

Register	Offset
Device_Capabilities_2_Register	94h

#### 28.4.2.51.2 Diagram



#### 28.4.2.51.3 Fields

Field	Function
31-6	Reserved

Table continues on the next page...

## Memory map/register overview

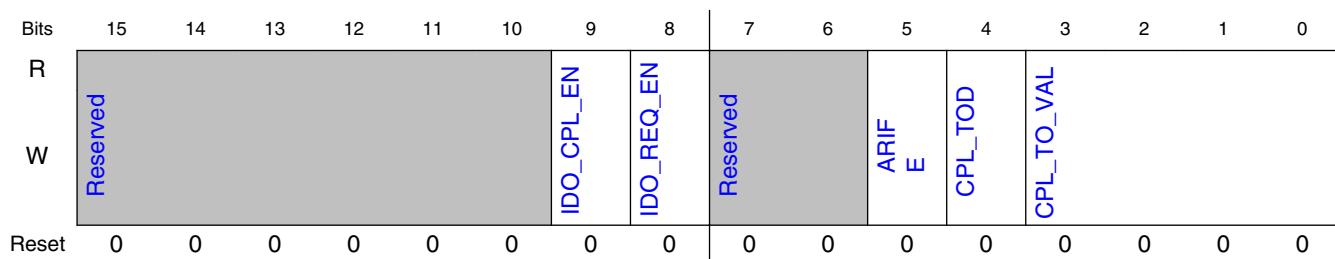
Field	Function
—	
5 ARI_FWD	ARI Forwarding Supported
4 CPL_TO_DS	Completion Timeout Disable Supported
3-0 CPL_TO_RS	Completion Timeout Ranges Supported

## 28.4.2.52 PCI Express Device Control 2 Register (Device\_Control\_2\_Register)

### 28.4.2.52.1 Offset

Register	Offset
Device_Control_2_Register	98h

### 28.4.2.52.2 Diagram



### 28.4.2.52.3 Fields

Field	Function
15-10 —	Reserved
9 IDO_CPL_EN	IDO Completion Enable
8 IDO_REQ_EN	IDO Request Enable

Table continues on the next page...

Field	Function
7-6	Reserved
—	
5 ARIFE	ARI Forwarding Enable Must be set when RC is communicating with ARI devices. When set will allow RC to issue configuration type 0 cycles with device number != 0.
4 CPL_TOD	Completion Timeout Disable
3-0 CPL_TO_VAL	Completion Timeout Value

### 28.4.2.53 PCI Express Link Capabilities 2 Register (Link\_Capabilities\_2\_Register)

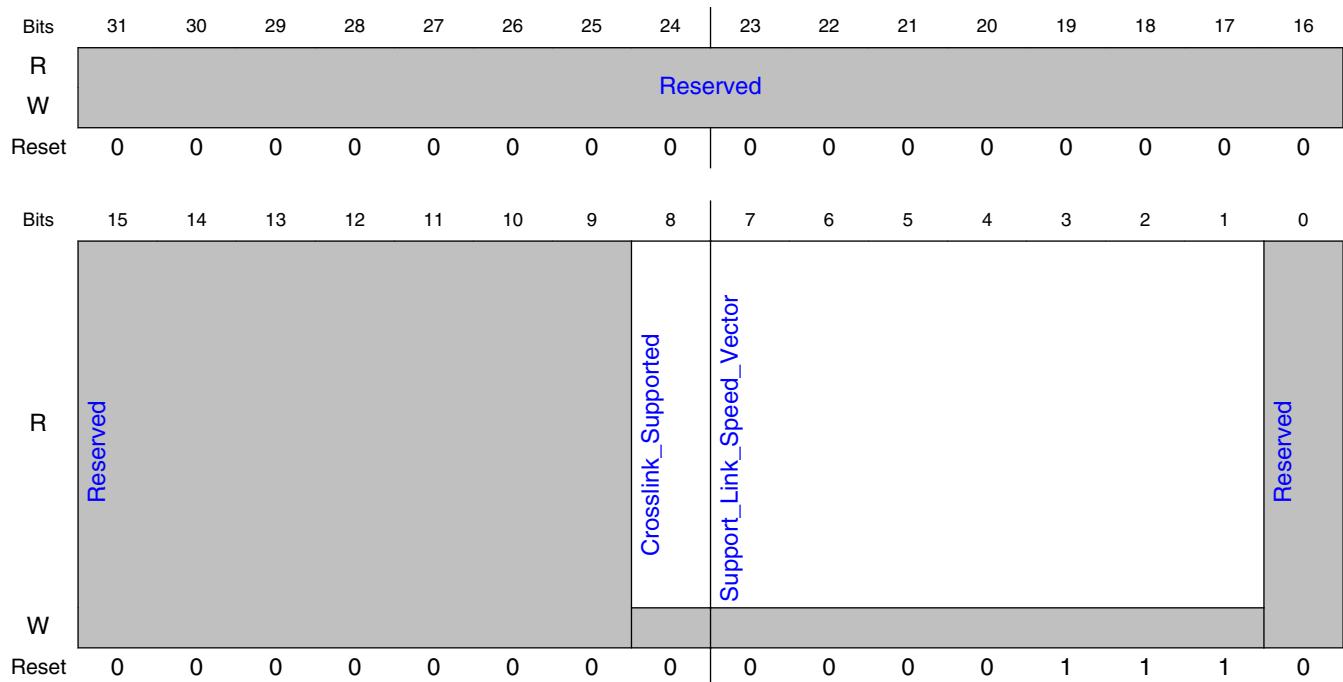
#### 28.4.2.53.1 Offset

Register	Offset
Link_Capabilities_2_Register	9Ch

#### 28.4.2.53.2 Function

The PCI Express link capability 2 register is shown below.

### 28.4.2.53.3 Diagram



### 28.4.2.53.4 Fields

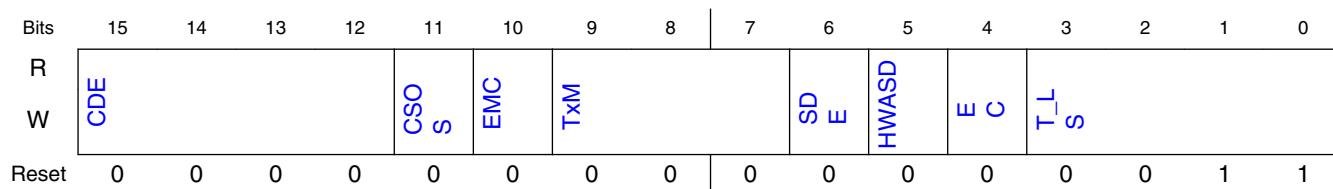
Field	Function
31-9 —	Reserved
8 Crosslink_Supp orted	Crosslink Supported
7-1 Support_Link_S peed_Vector	<p>Supported Link Speeds Vector</p> <p>This field indicates the supported Link speed(s) of the associated Port. For each bit, a value of 1 indicates that the corresponding Link speed is supported; otherwise, the Link speed is not supported. Bit definitions are:</p> <ul style="list-style-type: none"> <li>Bit 1 2.5 GT/s</li> <li>Bit 2 5.0 GT/s</li> <li>Bit 3 8.0 GT/s</li> <li>Bits 7:4 Reserved</li> </ul>
0 —	Reserved

## 28.4.2.54 PCI Express Link Control 2 Register (Link\_Control\_2\_Register)

### 28.4.2.54.1 Offset

Register	Offset
Link_Control_2_Register	A0h

### 28.4.2.54.2 Diagram



### 28.4.2.54.3 Fields

Field	Function
15-12	Compliance Preset/De-emphasis
CDE	
11	Compliance SOS
CSOS	
10	Enter Modified Compliance
EMC	
9-7	Transmit Margin
TxM	
6	Selectable De-emphasis
SDE	
5	Hardware Autonomous Speed Disable
HWASD	
4	Enter Compliance
EC	
3-0	Target Link Speed
T_LS	

## 28.4.2.55 PCI Express Link Status 2 Register (Link\_Status\_2\_Register)

### 28.4.2.55.1 Offset

Register	Offset
Link_Status_2_Register	A2h

### 28.4.2.55.2 Diagram



### 28.4.2.55.3 Fields

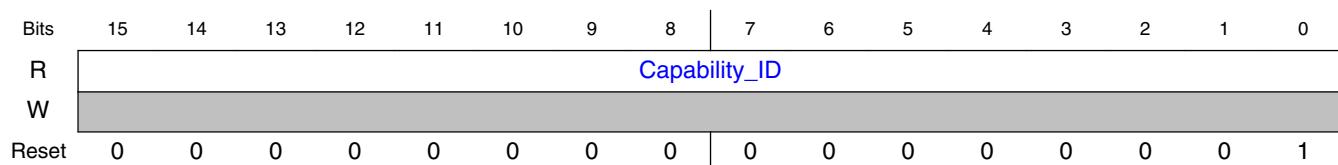
Field	Function
15-6 —	Reserved
5 LER	Link Equalization Request
4 EP3S	Equalization Phase 3 Successful
3 EP2S	Equalization Phase 2 Successful
2 EP1S	Equalization Phase 1 Successful
1 EC	Equalization Complete
0 DE_LVL	Current De-emphasis Level

## 28.4.2.56 PCI Express Advanced Error Reporting Capability ID Register (Advanced\_Error\_Reportin g\_Capability\_ID\_Regis ter)

### 28.4.2.56.1 Offset

Register	Offset
Advanced_Error_Reportin g_Capability_ID_Regis ter	100h

### 28.4.2.56.2 Diagram



### 28.4.2.56.3 Fields

Field	Function
15-0	PCI Express Extended Capability ID
Capability_ID	0x0001 = Advanced error reporting capability

## 28.4.2.57 PCI Express Uncorrectable Error Status Register (Uncorrectable\_Error\_Status\_Register)

### 28.4.2.57.1 Offset

Register	Offset
Uncorrectable_Error_Status_Register	104h

## 28.4.2.57.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved												UR E	ECRC E	MTLP	RXO	UC
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	CA	CTO	FCP E	PTLP	Reserved								DLP E	Reserved			
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	0	0	0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## 28.4.2.57.3 Fields

Field	Function
31-21	Reserved
—	
20 URE	Unsupported request error status.
19 ECRCE	ECRC error status.
18 MTLP	Malformed TLP status.
17 RXO	Receiver overflow status.
16 UC	Unexpected completion status.
15 CA	Completer abort status.
14 CTO	Completion timeout status. Note that a completion timeout error is a fatal error. If a completion timeout error is detected, the system has become unstable. Hot reset is recommended to restore stability of the system.
13 FCPE	Flow control protocol error status.
12 PTLP	Poisoned TLP status.
11-5 —	Reserved

Table continues on the next page...

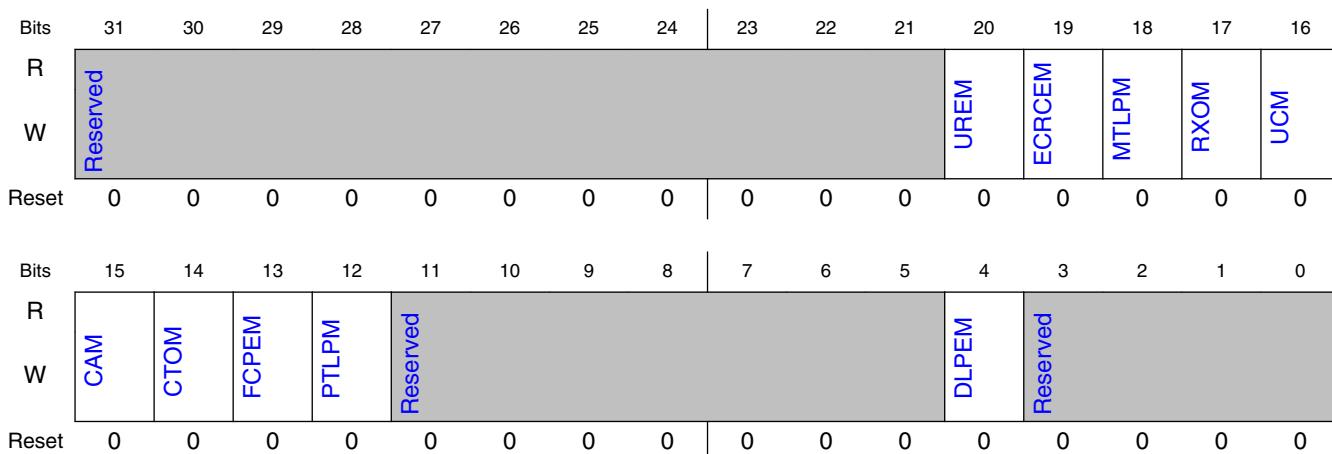
Field	Function
4 DLPE	Data link protocol error status.
3-0 —	Reserved

### 28.4.2.58 PCI Express Uncorrectable Error Mask Register (Uncorrectable\_Error\_Mask\_Register)

#### 28.4.2.58.1 Offset

Register	Offset
Uncorrectable_Error_Mask_Register	108h

#### 28.4.2.58.2 Diagram



#### 28.4.2.58.3 Fields

Field	Function
31-21 —	Reserved
20 UREM	Unsupported request error mask.

Table continues on the next page...

## Memory map/register overview

Field	Function
19 ECRCEM	ECRC error mask.
18 MTLPM	Malformed TLP mask.
17 RXOM	Receiver overflow mask.
16 UCM	Unexpected completion mask.
15 CAM	Completer abort mask.
14 CTOM	Completion timeout mask.
13 FCPEM	Flow control protocol error mask.
12 PTLPM	Poisoned TLP mask.
11-5 —	Reserved
4 DLPEM	Data link protocol error mask.
3-0 —	Reserved

### 28.4.2.59 PCI Express Uncorrectable Error Severity Register (Uncorrectable\_Error\_Severity\_Register)

#### 28.4.2.59.1 Offset

Register	Offset
Uncorrectable_Error_Severity_Register	10Ch

## 28.4.2.59.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved												URE S	ECRCE S	MTLPS	RXOS	UCS
W	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	CAS	CTOS	FCPE S	PTLP S	Reserved				DLPE S	Reserved							
W	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	
Reset	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	

## 28.4.2.59.3 Fields

Field	Function
31-21 —	Reserved
20 URES	Unsupported request error severity.
19 ECRCES	ECRC error severity.
18 MTLPS	Malformed TLP severity.
17 RXOS	Receiver overflow severity.
16 UCS	Unexpected completion severity.
15 CAS	Completer abort severity.
14 CTOS	Completion timeout severity.
13 FCPES	Flow control protocol error severity.
12 PTLPS	Poisoned TLP severity.
11-5 —	Reserved
4	Data link protocol error severity.

Table continues on the next page...

## Memory map/register overview

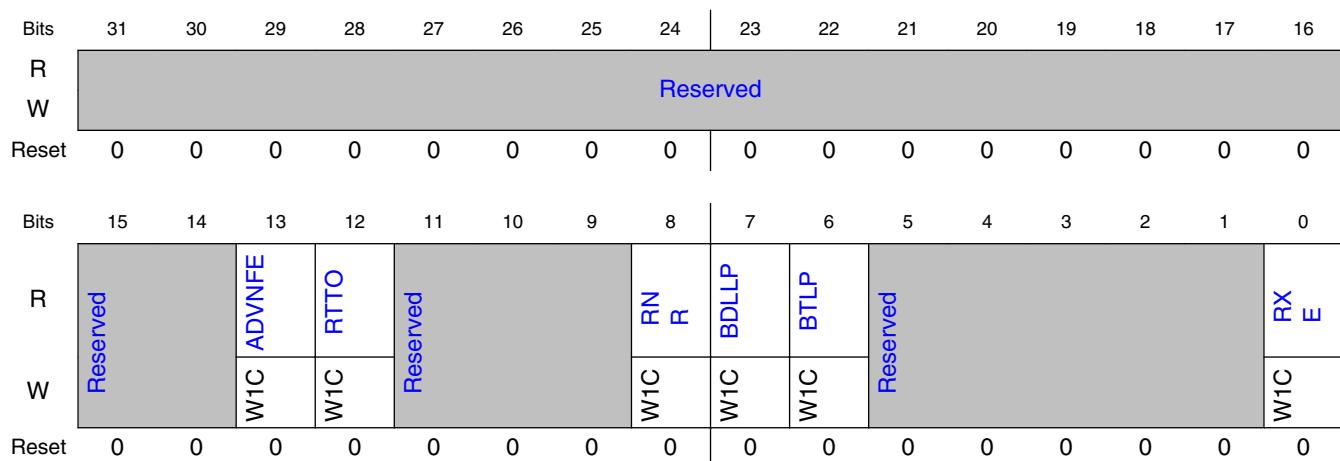
Field	Function
DLPES	
3-0	Reserved
—	

## 28.4.2.60 PCI Express Correctable Error Status Register (Correctable\_Error\_Status\_Register)

### 28.4.2.60.1 Offset

Register	Offset
Correctable_Error_Status_Register	110h

### 28.4.2.60.2 Diagram



### 28.4.2.60.3 Fields

Field	Function
31-14	Reserved
—	
13 ADVNFE	Advisory Non-Fatal Error Status indicates the occurrence of the advisory error, and the Advisory Non-Fatal Error Mask bit in the <a href="#">PCI Express Correctable Error Mask Register (Correctable_Error_Mask_Register)</a> is checked to determine whether to proceed further with logging and signaling
12	Replay timer timeout status

Table continues on the next page...

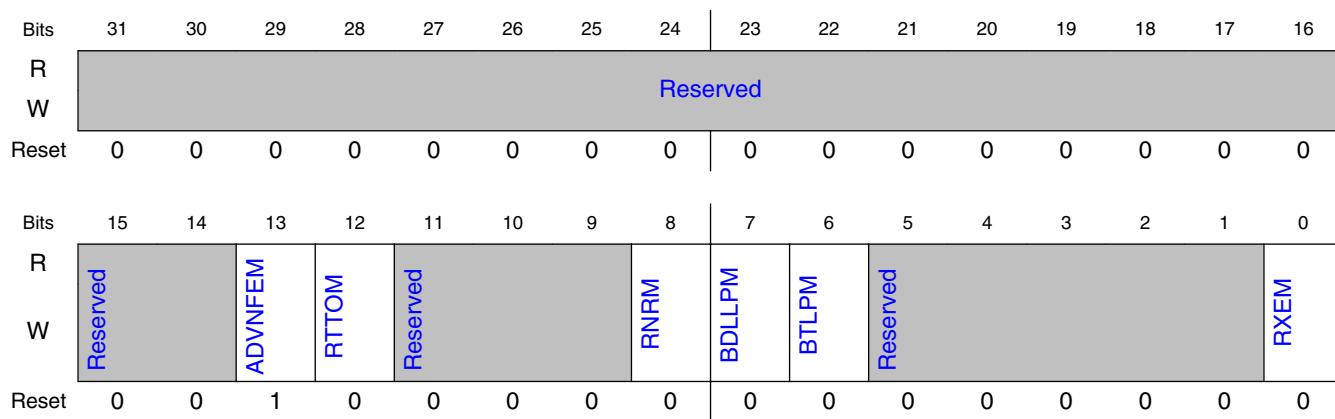
Field	Function
RTTO	
11-9 —	Reserved
8 RNR	REPLAY_NUM Rollover status
7 BDLLP	Bad DLLP status
6 BTLP	Bad TLP status
5-1 —	Reserved
0 RXE	Receiver error status

## 28.4.2.61 PCI Express Correctable Error Mask Register (Correctable\_Error\_Mask\_Register)

### 28.4.2.61.1 Offset

Register	Offset
Correctable_Error_Mask_Register	114h

### 28.4.2.61.2 Diagram



### 28.4.2.61.3 Fields

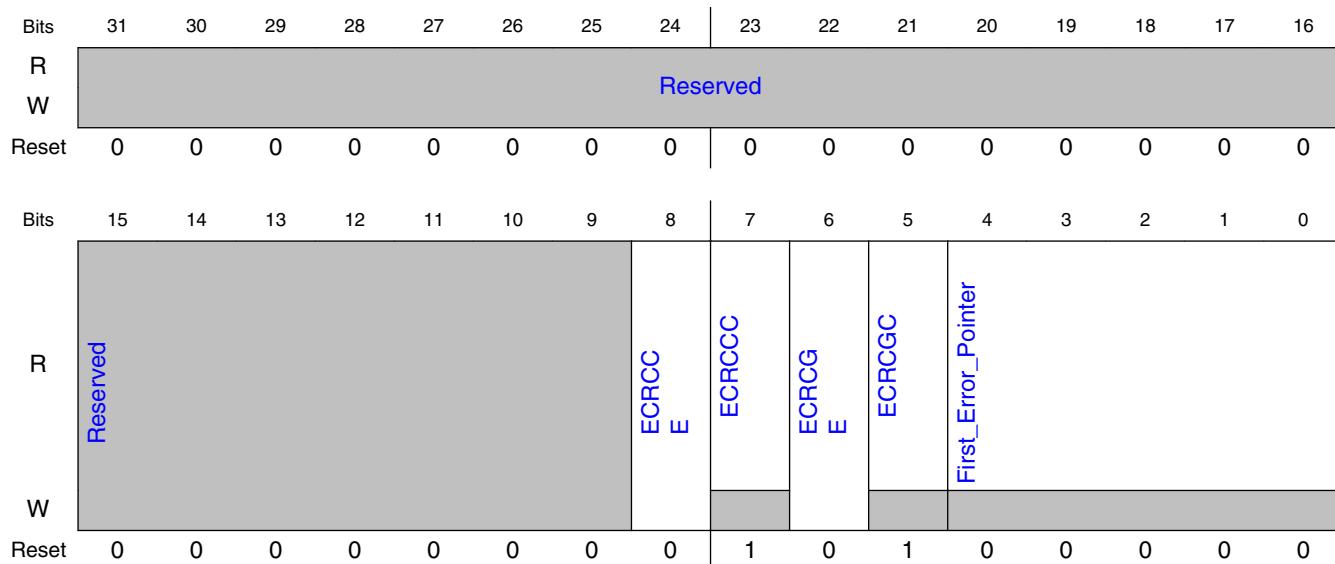
Field	Function
31-14 —	Reserved
13 ADVNFEM	Advisory non-fatal error mask. This bit is Set by default to enable compatibility with software that does not comprehend Role-Based Error Reporting
12 RTTOM	Replay timer timeout mask
11-9 —	Reserved
8 RNRM	REPLAY_NUM Rollover mask
7 BDLLPM	Bad DLLP mask
6 BTLPBM	Bad TLP mask
5-1 —	Reserved
0 RXEM	Receiver error mask

## 28.4.2.62 PCI Express Advanced Error Capabilities and Control Register (Advanced\_Error\_Capabilities\_and\_Control\_Register)

### 28.4.2.62.1 Offset

Register	Offset
Advanced_Error_Capabilities_and_Control_Register	118h

### 28.4.2.62.2 Diagram



### 28.4.2.62.3 Fields

Field	Function
31-9 —	Reserved
8 ECRCCE	ECRC checking enable.
7 ECRCCC	ECRC checking capable.
6 ECRCGE	ECRC generation enable.
5 ECRCGC	ECRC generation capable.
4-0 First_Error_Pointer	The First Error Pointer is a read-only register that identifies the bit position of the first error reported in the Uncorrectable Error Status register.

### 28.4.2.63 PCI Express Header Log Register 1 (Header\_Log\_Register\_DWORD1)

### 28.4.2.63.1 Offset

Register	Offset
Header_Log_Register_DWORD1	11Ch

### 28.4.2.63.2 Function

The first DWORD of the PCI Express header log register is shown in the figure below.

### 28.4.2.63.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.63.4 Fields

Field	Function
31-24 Byte_0	Byte n of the TLP header associated with the error.
23-16 Byte_1	Byte n of the TLP header associated with the error.
15-8 Byte_2	Byte n of the TLP header associated with the error.
7-0 Byte_3	Byte n of the TLP header associated with the error.

### 28.4.2.64 PCI Express Header Log Register 2 (Header\_Log\_Register\_DWORD2)

### 28.4.2.64.1 Offset

Register	Offset
Header_Log_Register_DWORD2	120h

### 28.4.2.64.2 Function

The second DWORD of the PCI Express header log register is shown in the figure below.

### 28.4.2.64.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																Byte_4
W																Byte_5
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																Byte_6
W																Byte_7
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.64.4 Fields

Field	Function
31-24 Byte_4	Byte n of the TLP header associated with the error.
23-16 Byte_5	Byte n of the TLP header associated with the error.
15-8 Byte_6	Byte n of the TLP header associated with the error.
7-0 Byte_7	Byte n of the TLP header associated with the error.

### 28.4.2.65 PCI Express Header Log Register 3 (Header\_Log\_Register\_DWORD3)

### 28.4.2.65.1 Offset

Register	Offset
Header_Log_Register_DWORD3	124h

### 28.4.2.65.2 Function

The third DWORD of the PCI Express header log register is shown in the figure below.

### 28.4.2.65.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.65.4 Fields

Field	Function
31-24 Byte_8	Byte n of the TLP header associated with the error.
23-16 Byte_9	Byte n of the TLP header associated with the error.
15-8 Byte_A	Byte n of the TLP header associated with the error.
7-0 Byte_B	Byte n of the TLP header associated with the error.

### 28.4.2.66 PCI Express Header Log Register 4 (Header\_Log\_Register\_DWORD4)

### 28.4.2.66.1 Offset

Register	Offset
Header_Log_Register_DWORD4	128h

### 28.4.2.66.2 Function

The fourth DWORD of the PCI Express header log register is shown in the figure below.

### 28.4.2.66.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.66.4 Fields

Field	Function
31-24 Byte_C	Byte n of the TLP header associated with the error.
23-16 Byte_D	Byte n of the TLP header associated with the error.
15-8 Byte_E	Byte n of the TLP header associated with the error.
7-0 Byte_F	Byte n of the TLP header associated with the error.

## 28.4.2.67 PCI Express Root Error Command Register (Root\_Error\_Command\_Register)

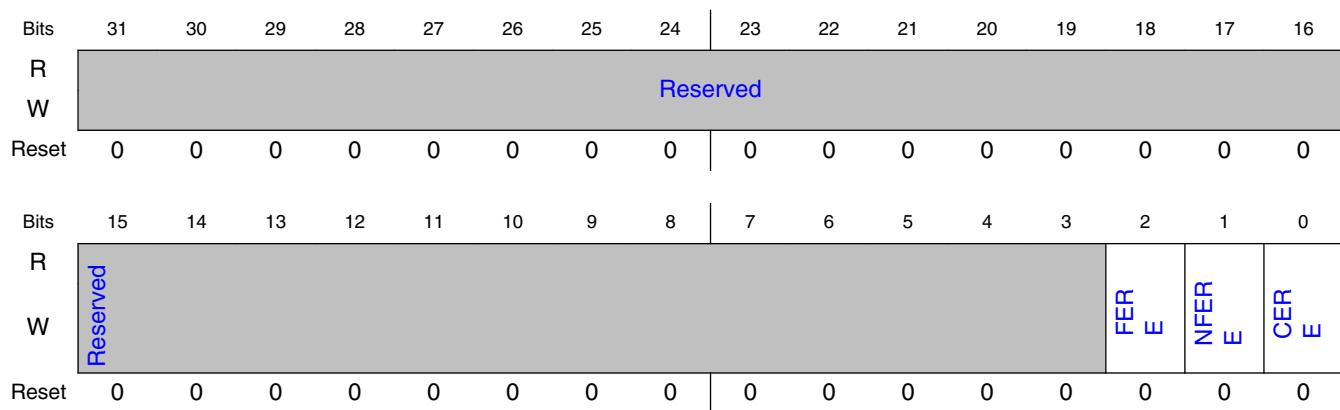
### 28.4.2.67.1 Offset

Register	Offset
Root_Error_Command_Register	12Ch

### 28.4.2.67.2 Function

This register is supported only for RC mode.

### 28.4.2.67.3 Diagram



### 28.4.2.67.4 Fields

Field	Function
31-3	Reserved
—	
2 FERE	Fatal error reporting enable.
1 NFERE	Non-fatal error reporting enable
0 CERE	Correctable error reporting enable

## 28.4.2.68 PCI Express Root Error Status Register (Root\_Error\_Status\_Register)

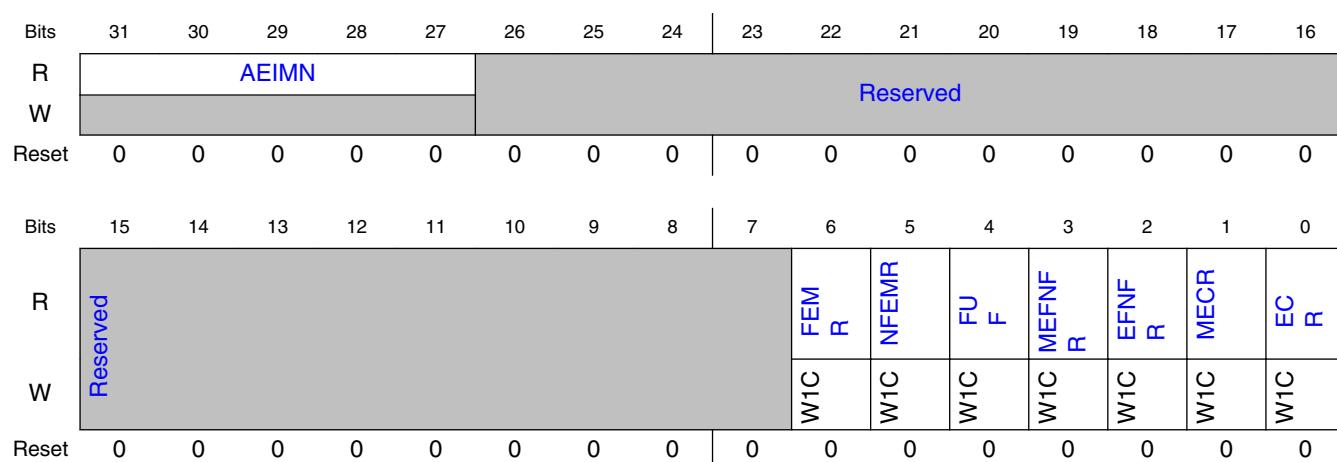
### 28.4.2.68.1 Offset

Register	Offset
Root_Error_Status_Register	130h

### 28.4.2.68.2 Function

This register is supported only for RC mode.

### 28.4.2.68.3 Diagram



### 28.4.2.68.4 Fields

Field	Function
31-27 AEIMN	Advanced error interrupt message number.
26-7 —	Reserved
6 FEMR	Fatal error messages received.
5 NFEMR	Non-fatal error messages received.

Table continues on the next page...

## Memory map/register overview

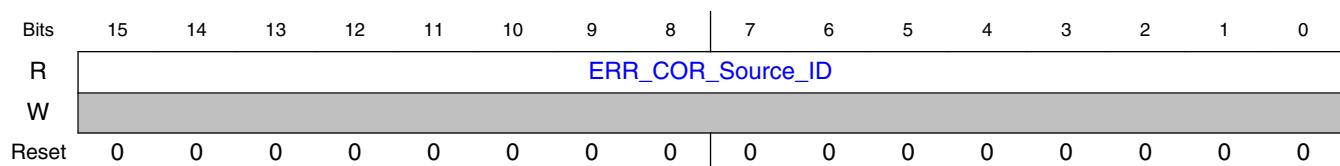
Field	Function
4 FUF	First uncorrectable fatal.
3 MEFNFR	Multiple ERR_FATAL/NONFATAL received.
2 EFNFR	ERR_FATAL/NONFATAL received.
1 MECR	Multiple ERR_COR received.
0 ECR	ERR_COR received.

## 28.4.2.69 PCI Express Correctable Error Source ID Register (Correctable\_Error\_Source\_ID\_Register)

### 28.4.2.69.1 Offset

Register	Offset
Correctable_Error_Source_ID_Register	134h

### 28.4.2.69.2 Diagram



### 28.4.2.69.3 Fields

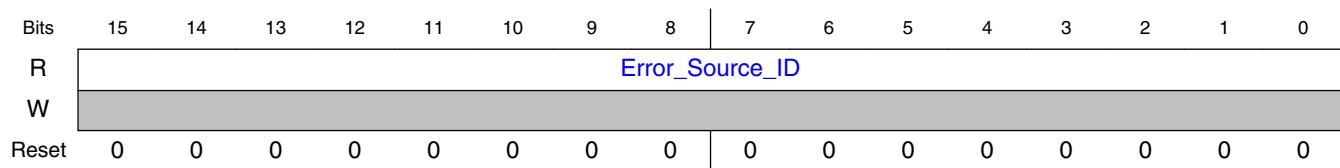
Field	Function
15-0 ERR_COR_Source_ID	Loaded with the requester ID indicated in the received ERR_COR Message when the ERR_COR Received register is not already set. Default value of this field is 0.

## 28.4.2.70 PCI Express Error Source ID Register (Error\_Source\_ID\_Register)

### 28.4.2.70.1 Offset

Register	Offset
Error_Source_ID_Register	136h

### 28.4.2.70.2 Diagram



### 28.4.2.70.3 Fields

Field	Function
15-0 Error_Source_ID	ERR_FATAL/NONFATAL source ID

## 28.4.2.71 Secondary PCI Express Extended Capability Header (SPCI\_E\_CAP\_HEADER\_REG)

### 28.4.2.71.1 Offset

Register	Offset
SPCIE_CAP_HEADER_REG	148h

## 28.4.2.71.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	NEXT_OFFSET												CAP_VERSION				
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	1
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	EXTENDED_CAP_ID																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	1	1	0	0	1

## 28.4.2.71.3 Fields

Field	Function
31-20 NEXT_OFFSET	Next Capability Offset  This field is a pointer to the next capability structure.  <b>NOTE:</b> The reset value of this field depends on the instantiation of the module. See <a href="#">PCI Express configuration registers</a> for the specific values.
19-16 CAP_VERSION	Capability Version  Indicates the version of the Capability structure present.
15-0 EXTENDED_CA_P_ID	PCI Express Extended Capability ID  0019h = Secondary PCI Express extended capability

## 28.4.2.72 Link Control 3 Register (LINK\_CONTROL3\_REG)

### 28.4.2.72.1 Offset

Register	Offset
LINK_CONTROL3_REG	14Ch

### 28.4.2.72.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														EQ_REQ_INT_EN	PERFORM_EQ
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.72.3 Fields

Field	Function
31-2	Reserved
—	
1	Link Equalization Request Interrupt Enable
EQ_REQ_INT_EN	This bit is RW for Downstream Ports only.
0	Perform Equalization
PERFORM_EQ	This bit is RW for Downstream Ports only.

## 28.4.2.73 Lane Error Status Register (LANE\_ERR\_STATUS\_REG)

### 28.4.2.73.1 Offset

Register	Offset
LANE_ERR_STATUS_REG	150h

### 28.4.2.73.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.73.3 Fields

Field	Function								
31-4	Reserved								
—									
3-0	Lane Error Status								
LANE_ERR_ST ATUS	Each bit indicates if the corresponding Lane detected a Lane-based error.  NOTE: This field is not supported in every instance. The following table includes only supported registers.								
<table border="1"> <thead> <tr> <th>Field supported in</th> <th>Field not supported in</th> </tr> </thead> <tbody> <tr> <td>PEX1_LANE_ERR_STATUS_REG</td> <td>—</td></tr> <tr> <td>PEX2_LANE_ERR_STATUS_REG[1-0]</td> <td>PEX2_LANE_ERR_STATUS_REG[3-2]</td></tr> <tr> <td>PEX3_LANE_ERR_STATUS_REG[1-0]</td> <td>PEX3_LANE_ERR_STATUS_REG[3-2]</td></tr> </tbody> </table>		Field supported in	Field not supported in	PEX1_LANE_ERR_STATUS_REG	—	PEX2_LANE_ERR_STATUS_REG[1-0]	PEX2_LANE_ERR_STATUS_REG[3-2]	PEX3_LANE_ERR_STATUS_REG[1-0]	PEX3_LANE_ERR_STATUS_REG[3-2]
Field supported in	Field not supported in								
PEX1_LANE_ERR_STATUS_REG	—								
PEX2_LANE_ERR_STATUS_REG[1-0]	PEX2_LANE_ERR_STATUS_REG[3-2]								
PEX3_LANE_ERR_STATUS_REG[1-0]	PEX3_LANE_ERR_STATUS_REG[3-2]								

### 28.4.2.74 Lane Equalization Control Register (LANE0\_EQUALIZATION\_CONTROL - LANE3\_EQUALIZATION\_CONTROL)

### 28.4.2.74.1 Offset

Register	Offset
LANE0_EQUALIZATION_CONTROL	154h
LANE1_EQUALIZATION_CONTROL	156h
LANE2_EQUALIZATION_CONTROL	158h
LANE3_EQUALIZATION_CONTROL	15Ah

### NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PEX1_LANE0_EQUALIZATION_CONTROL-LANE3_EQUALIZATION_CONTROL	—
PEX2_LANE0_EQUALIZATION_CONTROL-LANE1_EQUALIZATION_CONTROL	PEX2_LANE2_EQUALIZATION_CONTROL-LANE3_EQUALIZATION_CONTROL
PEX3_LANE0_EQUALIZATION_CONTROL-LANE1_EQUALIZATION_CONTROL	PEX3_LANE2_EQUALIZATION_CONTROL-LANE3_EQUALIZATION_CONTROL

### 28.4.2.74.2 Diagram



### 28.4.2.74.3 Register reset values

Register	Reset value
LANE0_EQUALIZATION_CONTROL–LANE1_EQUALIZATION_CONTROL	PEX1–PEX3: 7F7Fh
LANE2_EQUALIZATION_CONTROL–LANE3_EQUALIZATION_CONTROL	7F7Fh

### 28.4.2.74.4 Fields

Field	Function
15 —	Reserved
14-12 USP_RX_PRES ET_HINT	Upstream Port Receiver Preset Hint
11-8 USP_TX_PRES ET	Upstream Port Transmitter Preset
7 —	Reserved
6-4 DSP_RX_PRES ET_HINT	Downstream Port Receiver Preset Hint
3-0 DSP_TX_PRES ET	Downstream Port Transmitter Preset

### 28.4.2.75 Symbol Timer Register and Filter Mask 1 Register (SYMBOL\_TIMER\_FILTER\_1\_OFF)

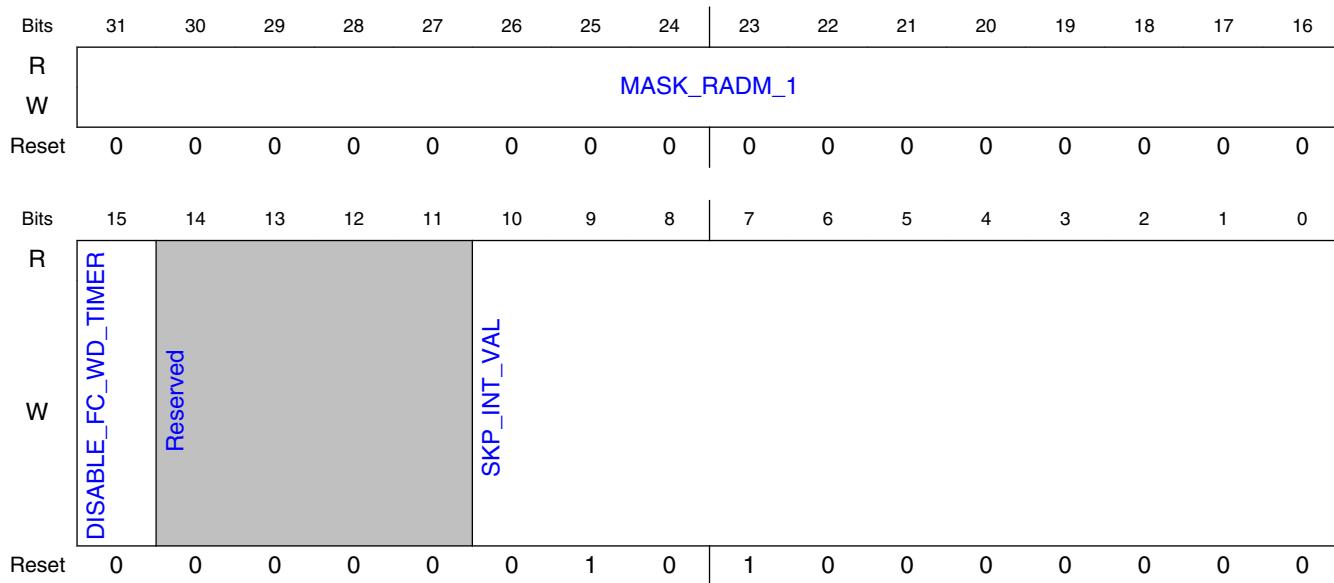
#### 28.4.2.75.1 Offset

Register	Offset
SYMBOL_TIMER_FILTER_1_OFF	71Ch

### 28.4.2.75.2 Function

The Filter Mask 1 Register modifies the receive filtering and error handling rules. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule

### 28.4.2.75.3 Diagram



### 28.4.2.75.4 Fields

Field	Function
MASK_RADM_1	<p>Filter Mask 1</p> <p>The Filter Mask 1 Register modifies the receive filtering and error handling rules. In each case, '0' applies the associated filtering rule and '1' masks the associated filtering rule.</p> <p>[31]: CX_FLT_MASK_RC_CFG_DISCARD 0: For RC filter to not allow CFG transaction being received 1: For RC filter to allow CFG transaction being received</p> <p>[30]: CX_FLT_MASK_RC_IO_DISCARD 0: For RC filter to not allow IO transaction being received 1: For RC filter to allow IO transaction being received</p> <p>[29]: CX_FLT_MASK_MSG_DROP 0: Drop MSG TLP (except for Vendor MSG). 1: Do not Drop MSG (except for Vendor MSG). Send message TLPs to your application.</p> <p>This bit only controls message TLPs other than Vendor MSGs. Vendor MSGs are controlled by Filter Mask Register 2, bits [1:0].</p> <p><b>NOTE:</b> The forwarding of message TLPs may be undesirable. If so, write a 0 to this bit to disable forwarding of message TLPs.</p> <p>[28]: CX_FLT_MASK_CPL_ECRC_DISCARD</p>

*Table continues on the next page...*

## Memory map/register overview

Field	Function
	<p>Only used when completion queue is advertized with infinite credits and is in store-and-forward mode.</p> <p>0: Discard completions with ECRC errors 1: Allow completions with ECRC errors to be passed up Reserved field for SW.</p> <p>[27]: CX_FLT_MASK_ECRC_DISCARD</p> <p>0: Discard TLPs with ECRC errors 1: Allow TLPs with ECRC errors to be passed up</p> <p>[26]: CX_FLT_MASK_CPL_LEN_MATCH</p> <p>0: Enforce length match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err 1: MASK length match for completions</p> <p>[25]: CX_FLT_MASK_CPL_ATTR_MATCH</p> <p>0: Enforce attribute match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask attribute match for completions</p> <p>[24]: CX_FLT_MASK_CPL_TC_MATCH</p> <p>0: Enforce Traffic Class match for completions; a violation results in a malformed TLP error, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask Traffic Class match for completions</p> <p>[23]: CX_FLT_MASK_CPL_FUNC_MATCH</p> <p>0: Enforce function match for completions; a violation results in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask function match for completions</p> <p>[22]: CX_FLT_MASK_CPL_REQID_MATCH</p> <p>0: Enforce Req. Id match for completions; a violation result in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask Req. Id match for completions</p> <p>[21]: CX_FLT_MASK_CPL_TAGERR_MATCH</p> <p>0: Enforce Tag Error Rules for completions; a violation result in cpl_abort, and possibly AER of unexp_cpl_err, cpl_rcvd_ur, cpl_rcvd_ca 1: Mask Tag Error Rules for completions</p> <p>[20]: CX_FLT_MASK_LOCKED_RD_AS_UR</p> <p>0: Treat locked Read TLPs as supported for RC. 1: Treat locked Read TLPs as UR for RC.</p> <p>[19]: CX_FLT_MASK_CFG_TYPE1_RE_AS_UR</p> <p>0: Treat CFG type1 TLPs as supported for RC. 1: Treat CFG type1 TLPs as UR for RC.</p> <p>[18]: CX_FLT_MASK_UR_OUTSIDE_BAR</p> <p>0: Treat out-of-bar TLPs as UR 1: Do not treat out-of-bar TLPs as UR</p> <p>[17]: CX_FLT_MASK_UR_POIS</p>

*Table continues on the next page...*

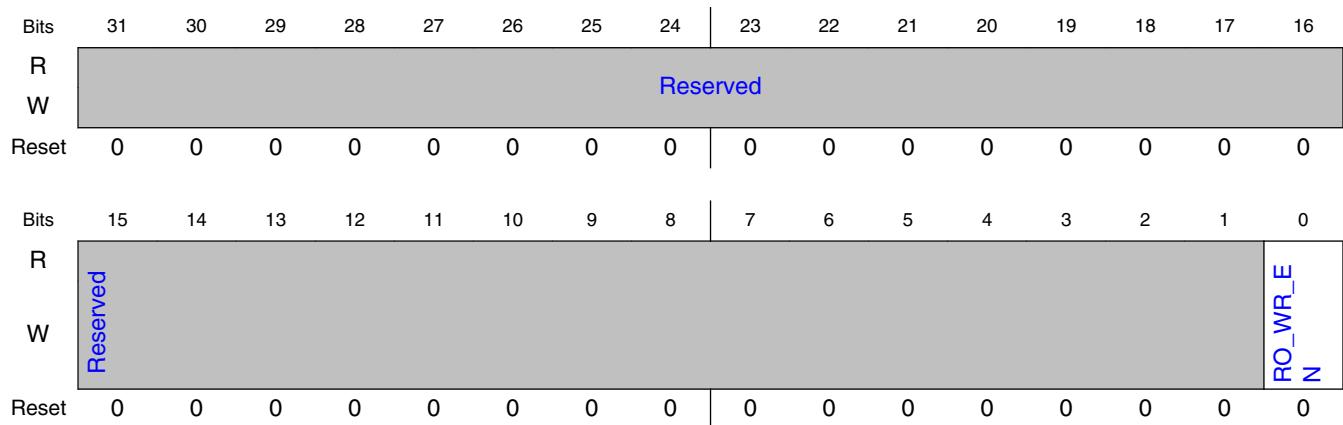
Field	Function
	<p>0: Treat poisoned request TLPs as UR      1: Do not treat poisoned request TLPs as UR      The native PEX controller always passes poisoned completions to your application.      [16]: CX_FLT_MASK_UR_FUNC_MISMATCH      0: Treat Function MisMatched TLPs as UR      1: Do not treat Function MisMatched TLPs as UR  <i>Note:</i> This register field is sticky.</p>
15 DISABLE_FC_WD_TIMER	<p>Disable FC Watchdog Timer      Disable FC Watchdog Timer.  <i>Note:</i> This register field is sticky.</p>
14-11 —	Reserved
10-0 SKP_INT_VAL	<p>SKP Interval Value      The number of symbol times to wait between transmitting SKP ordered sets. Note that the PEX controller actually waits the number of symbol times in this register plus 1 between transmitting SKP ordered sets. Your application must program this register accordingly. For example, if 1536 were programmed into this register (in a 250 MHz PEX controller), then the PEX controller actually transmits SKP ordered sets once every 1537 symbol times. The value programmed to this register is actually clock ticks and not symbol times. In a 125 MHz PEX controller, programming the value programmed to this register should be scaled down by a factor of 2 (because one clock tick = two symbol times in this case).  <i>Note:</i> This value is not used at Gen3 speed; the skip interval is hardcoded to 370 blocks.  <i>Note:</i> This register field is sticky.</p>

### 28.4.2.76 DBI Read-only Write Enable Register (MISC\_CONTROL\_1\_OFFSET)

#### 28.4.2.76.1 Offset

Register	Offset
MISC_CONTROL_1_OFF	8BCh

## 28.4.2.76.2 Diagram



## 28.4.2.76.3 Fields

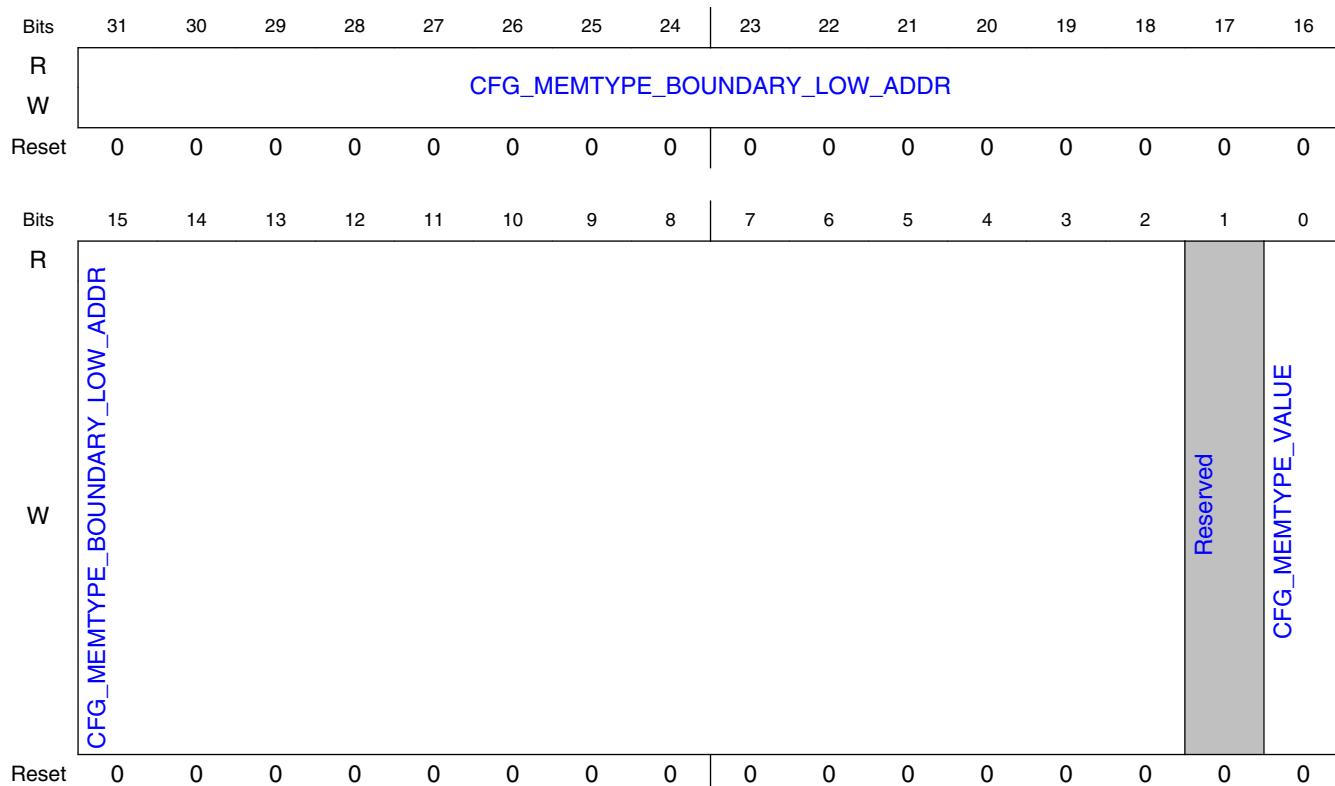
Field	Function
31-1 —	Reserved
0 RO_WR_EN	<p>Read-only write enable</p> <p>This field enables writing (using internal accesses only) to some read-only and hardware-initialized bits in the configuration registers.</p> <p>0b - Read-only fields are read only. 1b - Enables writing to some read-only fields</p>

## 28.4.2.77 Coherency Control Register 1 (COHERENCY\_CONTROL\_1\_L1\_OFFSET)

### 28.4.2.77.1 Offset

Register	Offset
COHERENCY_CONTROL_1_L1_OFFSET	8E0h

## 28.4.2.77.2 Diagram



## 28.4.2.77.3 Fields

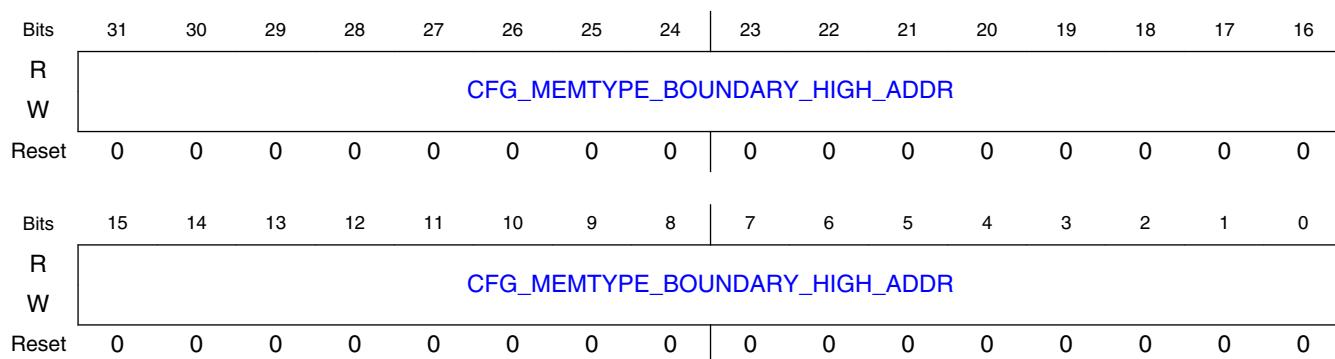
Field	Function
31-2 <code>CFG_MEMTYPE_BOUNDARY_LOW_ADDR</code>	Boundary lower address for memory type Bits [31:0] of dword-aligned address of the boundary for memory type. The two lower address least-significant bits must be 00. Addresses up to but not including this value are in the lower address space region; addresses equal or greater than this value are in the upper address space region. <i>Note:</i> This register field is sticky.
1 —	Reserved
0 <code>CFG_MEMTYPE_VALUE</code>	Memory type Sets the memory type for the lower and upper regions of the address space: <i>Note:</i> This register field is sticky. 0b - lower = CCSR; upper = Memory 1b - Reserved

## 28.4.2.78 Coherency Control Register 2 (COHERENCY\_CONTROL\_2\_OFF)

### 28.4.2.78.1 Offset

Register	Offset
COHERENCY_CONTROL_2_OFF	8E4h

### 28.4.2.78.2 Diagram



### 28.4.2.78.3 Fields

Field	Function
31-0 <code>CFG_MEMTYPE_BOUNDARY_HIGH_ADDR</code>	Boundary upper address for memory type Bits [63:32] of the 64-bit dword-aligned address of the boundary for memory type. Note: This register field is sticky.

## 28.4.2.79 Coherency Control Register 3 (COHERENCY\_CONTROL\_3\_OFF)

### 28.4.2.79.1 Offset

Register	Offset
COHERENCY_CONTROL_3_OFF	8E8h

## 28.4.2.79.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

## 28.4.2.79.3 Fields

Field	Function
31-0	Reserved Must be 0000_0000h.
—	

## 28.4.2.80 iATU Index Register (IATU\_VIEWPORT\_OFF)

### 28.4.2.80.1 Offset

Register	Offset
IATU_VIEWPORT_OFF	900h

### 28.4.2.80.2 Function

The iATU registers are programmed through an indirect addressing scheme (using this index register) to reduce the address footprint in the PCI Express extended configuration space.

The iATU can remap 6 outbound and 6 inbound address regions. This index register determines the memory region to be accessed.

### 28.4.2.80.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	REGION_DIR																
W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved								REGION_INDEX								
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 28.4.2.80.4 Fields

Field	Function
31 REGION_DIR	Region Direction Defines the region being accessed as either 0: Outbound 1: Inbound
30-8 —	Reserved
7-0 REGION_INDEX	Region Index Defines which region is being accessed when writing to the control, base, limit and target registers. Must not be set to a number greater than 5 when an outbound region is being accessed. Must not be set to a value greater than 5 when an inbound region is being accessed.

### 28.4.2.81 iATU Region Control 1 Register (IATU\_REGION\_CTRL\_1\_OFF\_INBOUND\_0)

#### 28.4.2.81.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_INBOUND_0	904h

### 28.4.2.81.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved				Reserved		Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	Reserved	Reserved				TYP E		
W										0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.81.3 Fields

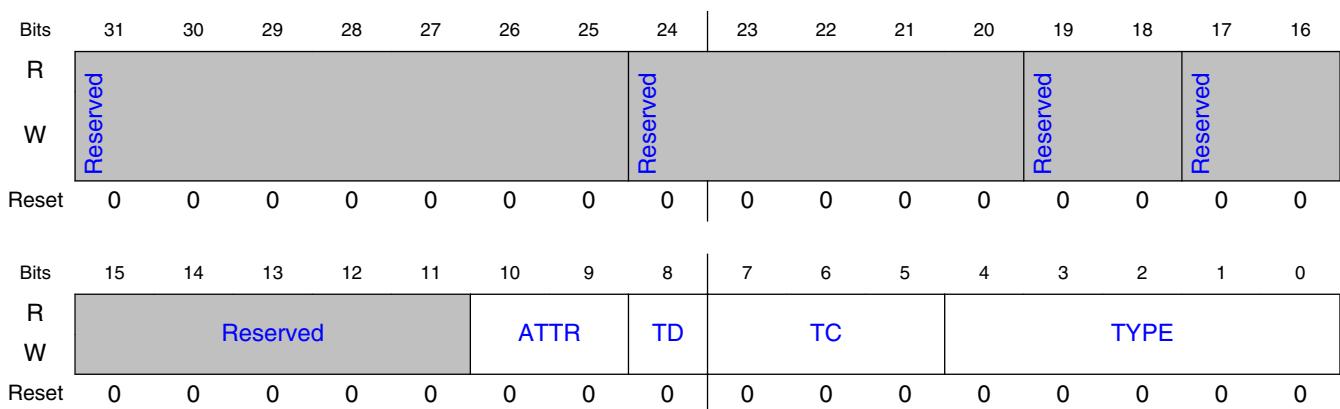
Field	Function
31-25	Reserved
—	
24-20	Reserved
—	
19-18	Reserved
—	
17-16	Reserved
—	
15-11	Reserved
—	
10-9	Reserved
—	
8	Reserved
—	
7-5	Reserved
—	
4-0	Type
TYPE	When the TYPE field of an inbound TLP is matched to this value, then address translation proceeds (when all other enabled field-matches are successful).

## 28.4.2.82 iATU Region Control 1 Register (IATU\_REGION\_CTRL\_1\_OFF\_OUTBOUND\_0)

### 28.4.2.82.1 Offset

Register	Offset
IATU_REGION_CTRL_1_OFF_OUTBOUND_0	904h

### 28.4.2.82.2 Diagram



### 28.4.2.82.3 Fields

Field	Function
31-25 —	Reserved
24-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-11 —	Reserved
10-9 ATTR	Attribute (Attr) When the address of an outbound TLP is matched to this region, then the ATTR field of the TLP is changed to the value in this register.
8	TLP Digest (TD)

Table continues on the next page...

Field	Function
TD	When the address of an outbound TLP is matched to this region, then the TD field of the TLP is changed to the value in this register.
7-5	Traffic class (TC)
TC	When the address of an outbound TLP is matched to this region, then the TC field of the TLP is changed to the value in this register.
4-0	Type
TYPE	When the address of an outbound TLP is matched to this region, then the TYPE field of the TLP is changed to the value in this register.

### 28.4.2.83 iATU Region Control 2 Register (IATU\_REGION\_CTRL\_2\_OFF\_INBOUND\_0)

#### 28.4.2.83.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_INBOUND_0	908h

#### 28.4.2.83.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REGION_N	MATCH_MODE	Reserved													
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved			Reserved			Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.83.3 Fields

Field	Function
31 REGION_EN	Region Enable This bit must be set to "1" for address translation to take place.
30 MATCH_MODE	Match Mode. Determines Inbound matching mode for TLPs. The mode depends on the type of TLP that is received as follows:  For MEM-I/O TLPs, this field is interpreted as follows: 0: Address Match Mode. The iATU operates using addresses as in the outbound direction. The Region Base and Limit Registers must be setup. 1:BAR Match Mode. BAR matching is used. The "BAR Number" field is relevant. Not used for RC.  For CFG0 TLPs, this field is interpreted as follows: 0: Routing ID Match Mode. The iATU interprets the Routing ID (Bytes 8 to 11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM-I/O transactions. The Routing ID of the TLP must be within the base and limit of the iATU region for matching to proceed. 1: Accept Mode. The iATU accepts all CFG0 transactions as address matches. The routing ID in the CFG0 TLP is ignored. This is useful as all received CFG0 TLPs should be processed regardless of the Bus number.  For MSG/MSGD TLPs, this field is interpreted as follows: 0: Address Match Mode. The iATU treats the third dword and fourth dword of the inbound MSG/MSGD TLP as an address and it is matched against the Region Base and Limit Registers. 1: Vendor ID Match Mode. This mode is relevant for ID-routed Vendor Defined Messages. The iATU ignores the Routing ID (Bus, Device, Function) in bits [31:16] of the third dword of the TLP header, but matches against the Vendor ID in bits [15:0] of the third dword of the TLP header. Bits [15:0] of the Region Upper Base register should be programmed with the required Vendor ID. The lower Base and Limit Register should be programmed to translate TLPs based on vendor specific information in the fourth dword of the TLP header.
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21 —	Reserved
20 —	Reserved

Table continues on the next page...

Field	Function
19	Reserved. Must be 0.
—	
18	Reserved
—	
17	Reserved
—	
16	Reserved
—	
15	Reserved
—	
14	Reserved
—	
13-11	Reserved
—	
10-8	Reserved. Must be 0.
—	
7-0	Reserved
—	

### 28.4.2.84 iATU Region Control 2 Register (IATU\_REGION\_CTRL\_2\_O FF\_OUTBOUND\_0)

#### 28.4.2.84.1 Offset

Register	Offset
IATU_REGION_CTRL_2_OFF_OUTBOUND_0	908h

## 28.4.2.84.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REGION_EN	Reserved	Reserved	CFG_SHIFT_MODE	Reserved	Reserved							Reserved	Reserved		
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 28.4.2.84.3 Fields

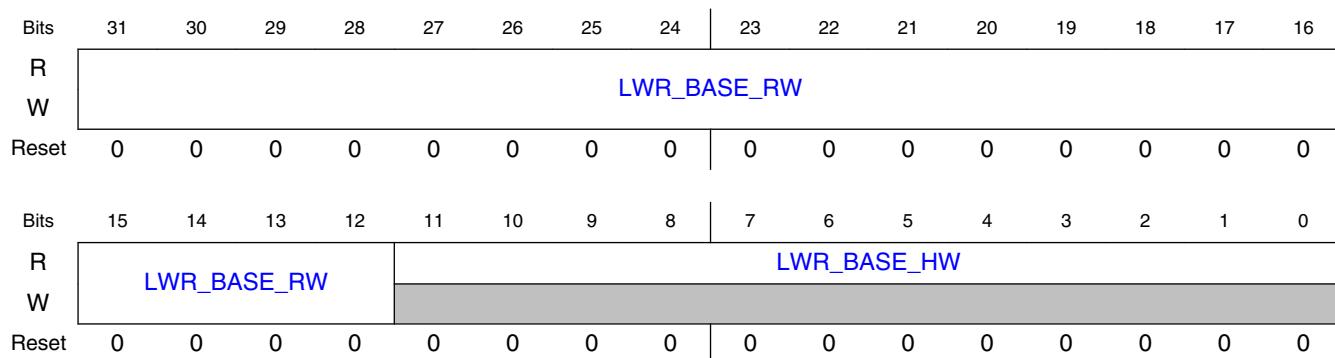
Field	Function
31	Region Enable
REGION_EN	This bit must be set to "1" for address translation to take place.
30	Reserved
—	
29	Reserved
—	
28	CFG Shift Mode
CFG_SHIFT_MODE	This is useful for CFG transactions where the PCIe configuration mechanism maps bits [27:12] of the address to the bus/device and function number. This allows a CFG configuration space to be located in any 256MB window of your application memory space using a 28-bit effective address. Shifts bits [27:12] of the untranslated address to form bits [31:16] of the translated address.
27	Reserved
—	
26-20	Reserved
—	
19	Reserved
—	
18-8	Reserved
—	
7-0	Message Code
MSG_CODE	Message TLPs: When the address of an outbound TLP is matched to this region, and the translated TLP TYPE field is Msg or MsgD; then the message field of the TLP is changed to the value in this register.

## 28.4.2.85 iATU Lower Base Address Register (IATU\_LWR\_BASE\_ADDRESS\_DR\_OFF\_INBOUND\_0)

### 28.4.2.85.1 Offset

Register	Offset
IATU_LWR_BASE_ADDRESS_DR_OFF_INBOUND_0	90Ch

### 28.4.2.85.2 Diagram



### 28.4.2.85.3 Fields

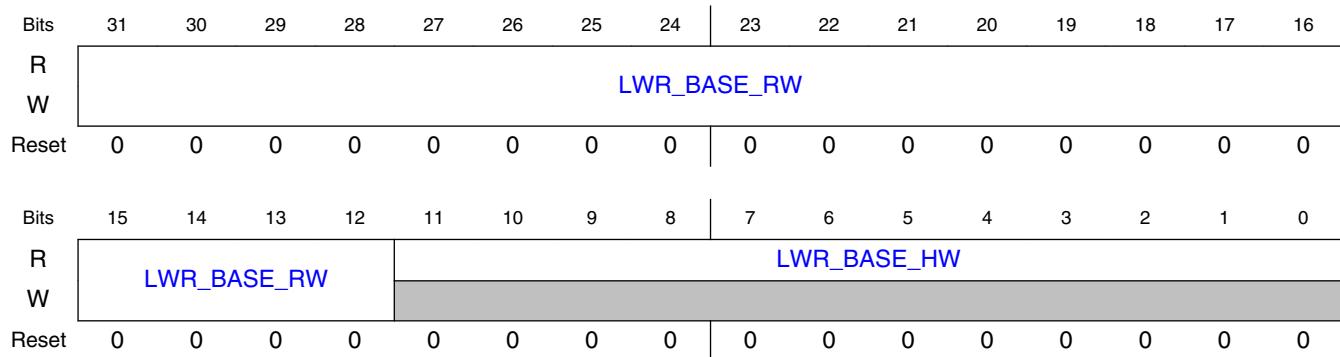
Field	Function
31-12 LWR_BASE_R W	Lower base address bits—programmable Forms bits [31:12] of the start address of the address region to be translated.
11-0 LWR_BASE_H W	Lower base address bits—hardwired Forms bits [11:0] of the start address of the address region to be translated. The start address must be aligned to a 4 KB boundary, so these bits are always 0. A write to this location is ignored by the PEX controller.

## 28.4.2.86 iATU Lower Base Address Register (IATU\_LWR\_BASE\_ADDRESS\_DR\_OFF\_OUTBOUND\_0)

### 28.4.2.86.1 Offset

Register	Offset
IATU_LWR_BASE_AD DR_OFF_OUTBOUND_0	90Ch

### 28.4.2.86.2 Diagram



### 28.4.2.86.3 Fields

Field	Function
31-12 LWR_BASE_R W	Lower base address bits—programmable Forms bits [31:12] of the start address of the address region to be translated.
11-0 LWR_BASE_H W	Lower base address bits—hardwired Forms bits [11:0] of the start address of the address region to be translated. The start address must be aligned to a 4 KB boundary, so these bits are always 0. A write to this location is ignored by the PEX controller.

## 28.4.2.87 iATU Upper Base Address Register (IATU\_UPPER\_BASE\_ADDR\_OFF\_INBOUND\_0)

### 28.4.2.87.1 Offset

Register	Offset
IATU_UPPER_BASE_ADDR_OFF_INBOUND_0	910h

### 28.4.2.87.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									UPPER_BASE_RW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									UPPER_BASE_RW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.87.3 Fields

Field	Function
31-0 UPPER_BASE_ RW	Upper base address bits Forms bits [63:32] of the start (and end) address of the address region to be translated.

### 28.4.2.88 iATU Upper Base Address Register (IATU\_UPPER\_BASE\_ADDR\_OFF\_OUTBOUND\_0)

#### 28.4.2.88.1 Offset

Register	Offset
IATU_UPPER_BASE_ ADDR_OFF_OUTBOUN D_0	910h

## 28.4.2.88.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	UPPER_BASE_RW																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	UPPER_BASE_RW																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

## 28.4.2.88.3 Fields

Field	Function
31-0	Upper base address bits
UPPER_BASE_RW	Forms bits [63:32] of the start (and end) address of the address region to be translated. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect.

## 28.4.2.89 iATU Limit Address Register (IATU\_LIMIT\_ADDR\_OFF\_INBOUND\_0)

### 28.4.2.89.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_INBOUND_0	914h

### 28.4.2.89.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LIMIT_ADDR_RW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LIMIT_ADDR_RW								LIMIT_ADDR_HW							
W	LIMIT_ADDR_RW															
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

### 28.4.2.89.3 Fields

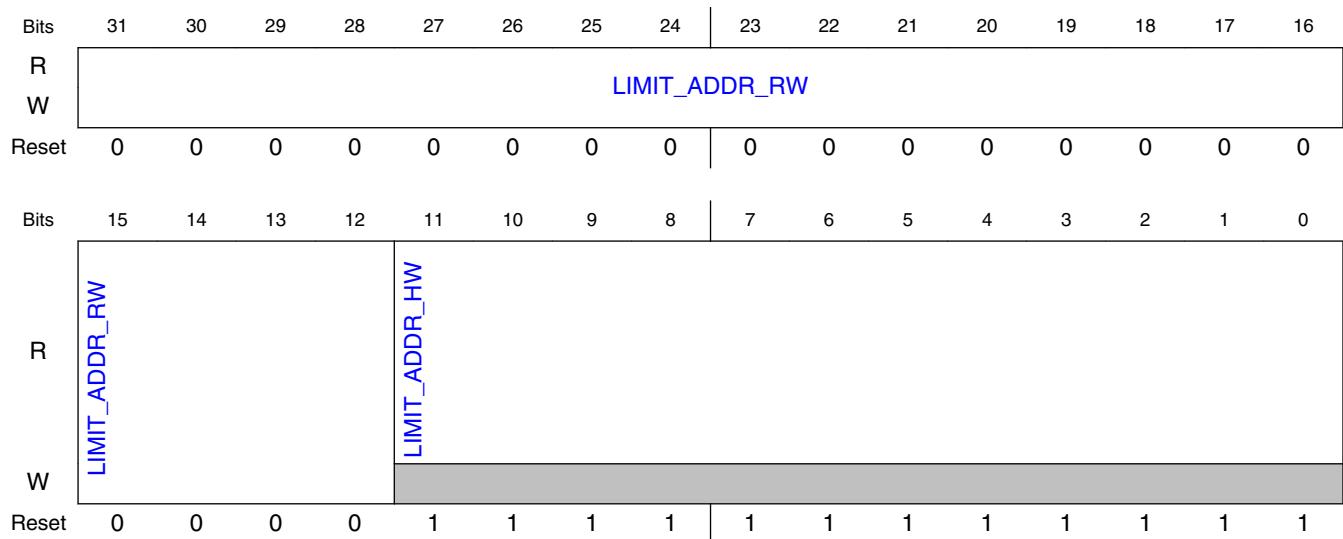
Field	Function
31-12 LIMIT_ADDR_R W	Limit address bits—programmable Forms bits [31:12] of the end address of the address region to be translated.
11-0 LIMIT_ADDR_H W	Limit address bits—hardwired Forms bits [11:0] of the end address of the address region to be translated. Address regions must be aligned to a 4 KB boundary, so these bits are always 1. A write to this location is ignored by the PEX controller.

### 28.4.2.90 iATU Limit Address Register (IATU\_LIMIT\_ADDR\_OFF\_OUTBOUND\_0)

#### 28.4.2.90.1 Offset

Register	Offset
IATU_LIMIT_ADDR_OFF_OUTBOUND_0	914h

## 28.4.2.90.2 Diagram



## 28.4.2.90.3 Fields

Field	Function
31-12 LIMIT_ADDR_R W	Limit address bits—programmable Forms bits [31:12] of the end address of the address region to be translated.
11-0 LIMIT_ADDR_H W	Limit address bits—hardwired Forms bits [11:0] of the end address of the address region to be translated. Address regions must be aligned to a 4 KB boundary, so these bits are always 1. A write to this location is ignored by the PEX controller.

## 28.4.2.91 iATU Region#N Lower Offset Address Register (IATU\_LWR\_TARGET\_ADDR\_OFF\_INBOUND\_0)

### 28.4.2.91.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_INBOUND_0	918h

### 28.4.2.91.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LWR_TARGET_RW															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LWR_TARGET_RW															
W	LWR_TARGET_HW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.4.2.91.3 Fields

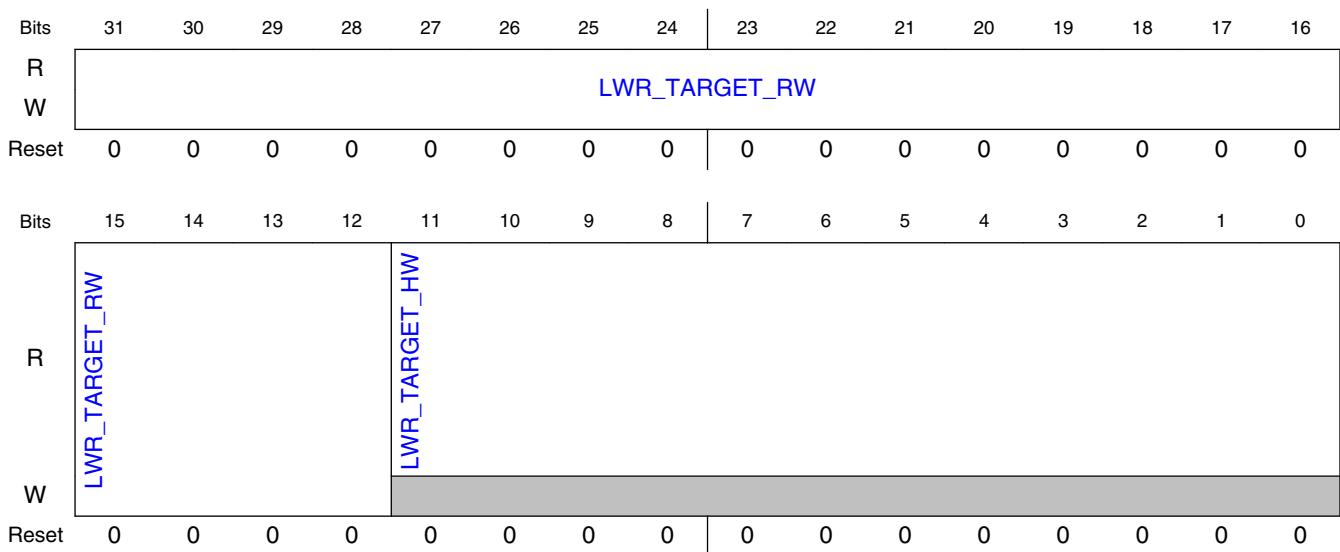
Field	Function
31-12 LWR_TARGET_RW	Lower target address bits—programmable Forms bits [31:12] of the new address of the translated region.
11-0 LWR_TARGET_HW	Lower target address bits—hardwired Forms bits [11:0] of the start address of the new address of the translated region. The start address must be aligned to a 4 KB boundary (in address match mode); and to the BAR size boundary (in BAR match mode) so these bits are always 0. If the BAR is smaller than the iATU region size, then the iATU target address must align to the iATU region size; otherwise it must align to the BAR size. A write to this location is ignored by the PEX controller.

## 28.4.2.92 iATU Outbound Region#N Lower Offset Address Register (IATU\_LWR\_TARGET\_ADDR\_OFF\_OUTBOUND\_0)

### 28.4.2.92.1 Offset

Register	Offset
IATU_LWR_TARGET_ADDR_OFF_OUTBOUND_0	918h

## 28.4.2.92.2 Diagram



## 28.4.2.92.3 Fields

Field	Function
31-12 LWR_TARGET_RW	Lower target address bits—programmable Forms bits [31:12] of the new address of the translated region.
11-0 LWR_TARGET_HW	Lower target address bits—hardwired Forms bits [11:0] of the start address of the new address of the translated region. The start address must be aligned to a 4 KB boundary, so these bits are always 0. A write to this location is ignored by the PEX controller.

## 28.4.2.93 iATU Upper Target Address Register (IATU\_UPPER\_TARGET\_ADDR\_OFF\_INBOUND\_0)

### 28.4.2.93.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_INBOUND_0	91Ch

### 28.4.2.93.2 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	UPPER_TARGET_RW																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	UPPER_TARGET_RW																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 28.4.2.93.3 Fields

Field	Function
31-0	Upper target address bits
UPPER_TARGET_RW	Forms bits [63:32] of the start address of the new address of the translated region. In systems with a 32-bit address space, this register is not used and therefore writing to this register has no effect.

## 28.4.2.94 iATU Upper Target Address Register (IATU\_UPPER\_TARGET\_ADDR\_OFF\_OUTBOUND\_0)

### 28.4.2.94.1 Offset

Register	Offset
IATU_UPPER_TARGET_ADDR_OFF_OUTBOUND_0	91Ch

## 28.4.2.94.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## 28.4.2.94.3 Fields

Field	Function
31-0	Upper target address bits
UPPER_TARGE T_RW	Forms bits [63:32] of the start address of the new address of the translated region.

## 28.4.2.95 Base Address Register 0 Mask (BAR0\_MASK)

### 28.4.2.95.1 Offset

Register	Offset
BAR0_MASK	1010h

### 28.4.2.95.2 Function

#### NOTE

This register is supported only on PEX1, PEX2, and PEX4.

PEX3 uses the configuration space 2 mechanism (via PEX\_LUT\_PEXLCTRL0[CS2]) to access the Base Address Register Masks.

The Base Address Register Mask is used to control size of memory requested by the corresponding PCI Express base address register (BAR) during initialization. When a mask bit is set to 1, the corresponding address bit in the BAR is masked and cannot be written, returning a 0 when read.

### 28.4.2.95.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W									MASK							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

### 28.4.2.95.4 Fields

Field	Function
31-1 MASK	Mask
0 BAR_EN	BAR enable. 0b - Disable BAR 1b - Enable BAR

## 28.4.2.96 Base Address Register 1 Mask (BAR1\_MASK)

### 28.4.2.96.1 Offset

Register	Offset
BAR1_MASK	1014h

### 28.4.2.96.2 Function

#### NOTE

This register is supported only on PEX1, PEX2, and PEX4. PEX3 uses the configuration space 2 mechanism (via PEX\_LUT\_PEXLCTRL0[CS2]) to access the Base Address Register Masks.

## Memory map/register overview

The Base Address Register Mask is used to control size of memory requested by the corresponding PCI Express base address register (BAR) during initialization. When a mask bit is set to 1, the corresponding address bit in the BAR is masked and cannot be written, returning a 0 when read.

### 28.4.2.96.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W									MASK							
Reset	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### 28.4.2.96.4 Fields

Field	Function
31-1 MASK	
0 BAR_EN	BAR enable. 0b - Disable BAR 1b - Enable BAR

### 28.4.2.97 Expansion ROM Base Address Register Mask (RC mode) (EXP\_ROM\_BAR\_MASK\_RC)

#### 28.4.2.97.1 Offset

Register	Offset
EXP_ROM_BAR_MASK_ RC	1038h

## 28.4.2.97.2 Function

### NOTE

This register is supported only on PEX1, PEX2, and PEX4. PEX3 uses the configuration space 2 mechanism (via PEX\_LUT\_PEXLCTRL0[CS2]) to access the Base Address Register Masks.

The Expansion ROM Base Address Register Mask is used to control size of memory requested by the corresponding PCI Express Expansion ROM Base Address Register (RC-Mode) during initialization. When a mask bit is set to 1, the corresponding address bit in the BAR is masked and cannot be written, returning a 0 when read.

## 28.4.2.97.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R																	
W	MASK																
Reset	0	0	0	0	0	0	0	0		1	1	1	1	1	1	1	1
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W	MASK																
Reset	1	1	1	1	1	1	1	1		1	1	1	1	1	1	1	0

## 28.4.2.97.4 Fields

Field	Function
31-11 MASK	Expansion ROM BAR Mask
10-1 —	Reserved
0 BAR_EN	Expansion ROM BAR enable. 0b - Disable Expansion ROM BAR 1b - Enable Expansion ROM BAR

## 28.5 PEX\_LUT memory map/registers overview

PEX\_LUT provides registers for:

- mapping Stream IDs to request IDs.
- generating a soft-reset
- determining the current LTSSM state

## 28.5.1 PEX\_LUT register descriptions

### 28.5.1.1 PEX\_LUT memory map

PEX1\_LUT base address: 341\_0000h

PEX2\_LUT base address: 351\_0000h

PEX3\_LUT base address: 361\_0000h

Offset	Register	Width (In bits)	Access	Reset value
20h	PEX LUT Status Register (PEXLSR)	32	W1C	0000_0000h
24h	PEX LUT Control Register (PEXLCR)	32	RW	0000_0000h
7FCh	PEX LUT Debug Register (PEXLDBG)	32	RW	0000_0000h
800h	PEX LUT Entry a Upper Data Register (PEXL0UDR)	32	RW	0000_FFFFh
804h	PEX LUT Entry a Lower Data Register (PEXL0LDR)	32	RW	8000_0000h
808h	PEX LUT Entry a Upper Data Register (PEXL1UDR)	32	RW	0000_0000h
80Ch	PEX LUT Entry a Lower Data Register (PEXL1LDR)	32	RW	0000_0000h
810h	PEX LUT Entry a Upper Data Register (PEXL2UDR)	32	RW	0000_0000h
814h	PEX LUT Entry a Lower Data Register (PEXL2LDR)	32	RW	0000_0000h
818h	PEX LUT Entry a Upper Data Register (PEXL3UDR)	32	RW	0000_0000h
81Ch	PEX LUT Entry a Lower Data Register (PEXL3LDR)	32	RW	0000_0000h
820h	PEX LUT Entry a Upper Data Register (PEXL4UDR)	32	RW	0000_0000h
824h	PEX LUT Entry a Lower Data Register (PEXL4LDR)	32	RW	0000_0000h
828h	PEX LUT Entry a Upper Data Register (PEXL5UDR)	32	RW	0000_0000h
82Ch	PEX LUT Entry a Lower Data Register (PEXL5LDR)	32	RW	0000_0000h
830h	PEX LUT Entry a Upper Data Register (PEXL6UDR)	32	RW	0000_0000h
834h	PEX LUT Entry a Lower Data Register (PEXL6LDR)	32	RW	0000_0000h
838h	PEX LUT Entry a Upper Data Register (PEXL7UDR)	32	RW	0000_0000h
83Ch	PEX LUT Entry a Lower Data Register (PEXL7LDR)	32	RW	0000_0000h
840h	PEX LUT Entry a Upper Data Register (PEXL8UDR)	32	RW	0000_0000h
844h	PEX LUT Entry a Lower Data Register (PEXL8LDR)	32	RW	0000_0000h
848h	PEX LUT Entry a Upper Data Register (PEXL9UDR)	32	RW	0000_0000h
84Ch	PEX LUT Entry a Lower Data Register (PEXL9LDR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
850h	PEX LUT Entry a Upper Data Register (PEXL10UDR)	32	RW	0000_0000h
854h	PEX LUT Entry a Lower Data Register (PEXL10LDR)	32	RW	0000_0000h
858h	PEX LUT Entry a Upper Data Register (PEXL11UDR)	32	RW	0000_0000h
85Ch	PEX LUT Entry a Lower Data Register (PEXL11LDR)	32	RW	0000_0000h
860h	PEX LUT Entry a Upper Data Register (PEXL12UDR)	32	RW	0000_0000h
864h	PEX LUT Entry a Lower Data Register (PEXL12LDR)	32	RW	0000_0000h
868h	PEX LUT Entry a Upper Data Register (PEXL13UDR)	32	RW	0000_0000h
86Ch	PEX LUT Entry a Lower Data Register (PEXL13LDR)	32	RW	0000_0000h
870h	PEX LUT Entry a Upper Data Register (PEXL14UDR)	32	RW	0000_0000h
874h	PEX LUT Entry a Lower Data Register (PEXL14LDR)	32	RW	0000_0000h
878h	PEX LUT Entry a Upper Data Register (PEXL15UDR)	32	RW	0000_0000h
87Ch	PEX LUT Entry a Lower Data Register (PEXL15LDR)	32	RW	0000_0000h
880h	PEX LUT Entry a Upper Data Register (PEXL16UDR)	32	RW	0000_0000h
884h	PEX LUT Entry a Lower Data Register (PEXL16LDR)	32	RW	0000_0000h
888h	PEX LUT Entry a Upper Data Register (PEXL17UDR)	32	RW	0000_0000h
88Ch	PEX LUT Entry a Lower Data Register (PEXL17LDR)	32	RW	0000_0000h
890h	PEX LUT Entry a Upper Data Register (PEXL18UDR)	32	RW	0000_0000h
894h	PEX LUT Entry a Lower Data Register (PEXL18LDR)	32	RW	0000_0000h
898h	PEX LUT Entry a Upper Data Register (PEXL19UDR)	32	RW	0000_0000h
89Ch	PEX LUT Entry a Lower Data Register (PEXL19LDR)	32	RW	0000_0000h
8A0h	PEX LUT Entry a Upper Data Register (PEXL20UDR)	32	RW	0000_0000h
8A4h	PEX LUT Entry a Lower Data Register (PEXL20LDR)	32	RW	0000_0000h
8A8h	PEX LUT Entry a Upper Data Register (PEXL21UDR)	32	RW	0000_0000h
8ACh	PEX LUT Entry a Lower Data Register (PEXL21LDR)	32	RW	0000_0000h
8B0h	PEX LUT Entry a Upper Data Register (PEXL22UDR)	32	RW	0000_0000h
8B4h	PEX LUT Entry a Lower Data Register (PEXL22LDR)	32	RW	0000_0000h
8B8h	PEX LUT Entry a Upper Data Register (PEXL23UDR)	32	RW	0000_0000h
8BCh	PEX LUT Entry a Lower Data Register (PEXL23LDR)	32	RW	0000_0000h
8C0h	PEX LUT Entry a Upper Data Register (PEXL24UDR)	32	RW	0000_0000h
8C4h	PEX LUT Entry a Lower Data Register (PEXL24LDR)	32	RW	0000_0000h
8C8h	PEX LUT Entry a Upper Data Register (PEXL25UDR)	32	RW	0000_0000h
8CCh	PEX LUT Entry a Lower Data Register (PEXL25LDR)	32	RW	0000_0000h
8D0h	PEX LUT Entry a Upper Data Register (PEXL26UDR)	32	RW	0000_0000h
8D4h	PEX LUT Entry a Lower Data Register (PEXL26LDR)	32	RW	0000_0000h
8D8h	PEX LUT Entry a Upper Data Register (PEXL27UDR)	32	RW	0000_0000h
8DCh	PEX LUT Entry a Lower Data Register (PEXL27LDR)	32	RW	0000_0000h
8E0h	PEX LUT Entry a Upper Data Register (PEXL28UDR)	32	RW	0000_0000h
8E4h	PEX LUT Entry a Lower Data Register (PEXL28LDR)	32	RW	0000_0000h
8E8h	PEX LUT Entry a Upper Data Register (PEXL29UDR)	32	RW	0000_0000h
8ECb	PEX LUT Entry a Lower Data Register (PEXL29LDR)	32	RW	0000_0000h

Table continues on the next page...

## PEX\_LUT memory map/registers overview

Offset	Register	Width (In bits)	Access	Reset value
8F0h	PEX LUT Entry a Upper Data Register (PEXL30UDR)	32	RW	0000_0000h
8F4h	PEX LUT Entry a Lower Data Register (PEXL30LDR)	32	RW	0000_0000h
8F8h	PEX LUT Entry a Upper Data Register (PEXL31UDR)	32	RW	0000_0000h
8FCh	PEX LUT Entry a Lower Data Register (PEXL31LDR)	32	RW	0000_0000h

## 28.5.1.2 PEX LUT Status Register (PEXLSR)

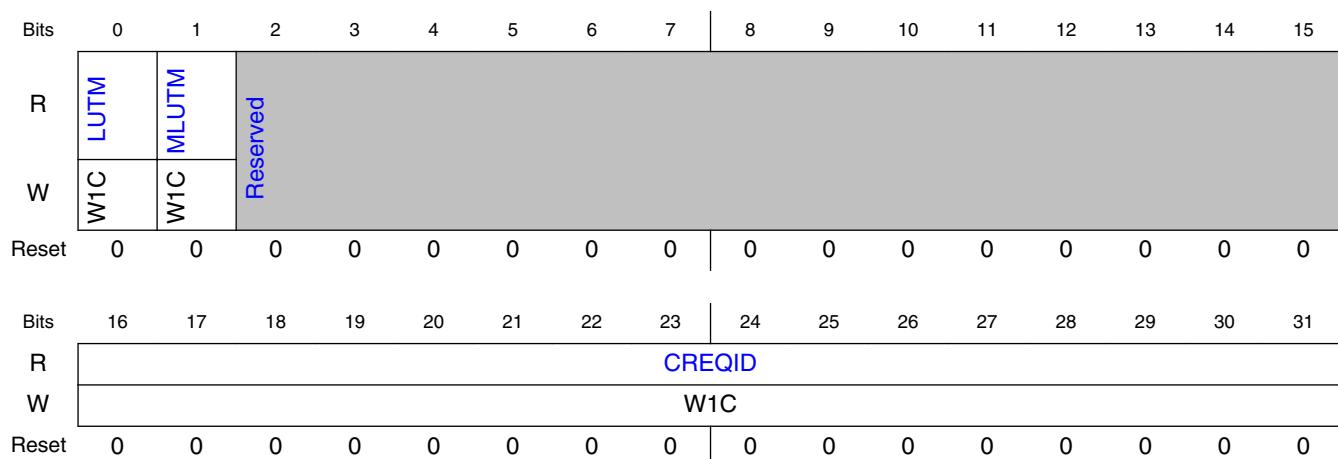
### 28.5.1.2.1 Offset

Register	Offset
PEXLSR	20h

### 28.5.1.2.2 Function

The PEX LUT Status Register (PEXLSR) logs the status of a lookup request, through the PEXLSR[LUTM] field. When no match is detected for all the lookup table entries, the lookup table miss LUTM field will be set. PEXLSR[LUTM] can be cleared by writing a 1 to it.

### 28.5.1.2.3 Diagram



### 28.5.1.2.4 Fields

Field	Function
0 LUTM	LUTM: Lookup table miss. This field is set when the AXI REQID does not match any of the lookup table entries. This field remains set from the time of an LUT miss until cleared by writing 1 to it.
1 MLUTM	MLUTM: Multiple lookup table miss. This field is set when more than one lookup failed. This field is set from the time of the second LUT miss until cleared by writing 1 to it. This bit is also set if a single miss occurs on both write and read channels simultaneously.
2-15 —	Reserved
16-31 CREQID	CREQID: Captured REQID. This field is the request ID of the first lookup table miss. If a simultaneous LUT miss is detected on both write and read channels, capture priority is given to the write channel REQID and multiple miss is reported in the MLUTM bit. This field remains unchanged from the time of an LUT miss until cleared by writing all 1s to it.

### 28.5.1.3 PEX LUT Control Register (PEXLCR)

#### 28.5.1.3.1 Offset

Register	Offset
PEXLCR	24h

#### 28.5.1.3.2 Function

The PEX LUT Control Register (PEXLCR) represents the Address Management Qualifier (AMQ) attributes, PL, BMT, 14'h0, ICID[14:0], assigned to the REQID translation when there is a lookup table miss.

### 28.5.1.3.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DPL	DBMT	Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W		DICID														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.5.1.3.4 Fields

Field	Function
0 DPL	DPL: Default privilege Level. This field is the default AMQ privilege level attribute for a lookup table miss.
1 DBMT	DBMT: Default bypass memory translation. This field is the default AMQ bypass memory translation attribute for a lookup table miss.
2-16 —	Reserved
17-31 DICID	DICID: Default isolation context ID. This field is the default AMQ isolation context ID attribute for a lookup table miss

## 28.5.1.4 PEX LUT Debug Register (PEXLDBG)

### 28.5.1.4.1 Offset

Register	Offset
PEXLDBG	7FCh

### 28.5.1.4.2 Function

The PEX LUT Debug Register (PEXLDBG) is used for debug purposes only. By setting PEXLDBG[WE] bit the user has access to the PEXLDBG[SR]. Once set the PEXLDBG[SR] will enable the soft reset to the PEX wrapper. PEXLDBG[LTSSM] is a read only field that reports the LTSSM status from the PEX block.

### 28.5.1.4.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	WE	SR	Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved												LTSSM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.5.1.4.4 Fields

Field	Function
0 WE	Write Enable This bit when set allows the user to have write access to the PEXDBG[SR] for both setting and clearing the soft reset on the PEX module.
1 SR	Soft Reset When SR is set to 1, the PEX module enters soft reset. The PEX mode remains in soft reset while SR is set to 1. When SR is cleared, the PEX module exits soft reset. If SR is to be set and cleared multiple times back to back, software must wait a few seconds before changing the bit value to allow each new state to take effect.
2-25 —	Reserved
26-31 LTSSM	Link Training Status State Machine (LTSSM) status This field is read only and captures the PEX block LTSSM state at each clock cycle. These bits indicate the link training status. This field is useful for debugging link training failures. <b>NOTE:</b> The status code changes while reacting to the link status. Therefore, software may need to read LTSSM multiple times to ensure the value has stabilized. 000000b - DETECT QUIET 000001b - DETECT ACTIVE 000010b - POLL ACTIVE 000011b - POLL COMPLIANCE 000100b - POLL CONFIG 000101b - PRE DETECT QUIET 000110b - DETECT WAIT

## PEX\_LUT memory map/registers overview

Field	Function
	000111b - CFG_LINKWD_START 001000b - CFG_LINKWD_ACEPT 001001b - CFG_LANENUM_WAIT 001010b - CFG_LANENUM_ACEPT 001011b - CFG_COMPLETE 001100b - CFG_IDLE 001101b - RCVRY_LOCK 001110b - RCVRY_SPEED 001111b - RCVRY_RCVRCFG 010000b - RCVRY_IDLE 010001b - L0 010010b - L0S 010011b - L123_SEND_EIDLE 010100b - L1_IDLE 010101b - L2_IDLE 010110b - L2_WAKE 010111b - DISABLED_ENTRY 011000b - DISABLED_IDLE 011001b - DISABLED 011010b - LPBK_ENTRY 011011b - LPBK_ACTIVE 011100b - LPBK_EXIT 011101b - LPBK_EXIT_TIMEOUT 011110b - HOT_RESET_ENTRY 011111b - HOT_RESET 100000b - RCVRY_EQ0 100001b - RCVRY_EQ1 100010b - RCVRY_EQ2 100011b - RCVRY_EQ3

### 28.5.1.5 PEX LUT Entry a Upper Data Register (PEXL0UDR)

#### 28.5.1.5.1 Offset

Register	Offset
PEXL0UDR	800h

#### 28.5.1.5.2 Function

The PEX LUT Entry  $a$  Upper Data Register (PEXL $a$ UDR) represents the upper data fields of entry  $a$  within the PEX lookup table.

### 28.5.1.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									REQID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									MASK							
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### 28.5.1.5.4 Fields

Field	Function
0-15 REQID	REQID: The Request ID of entry <i>a</i> of the PEX lookup table. This field is qualified by the MASK field PEXLaUDR[MASK] of the same PEX lookup table entry when attempting to match.
16-31 MASK	MASK: The mask field of entry <i>a</i> of the PEX lookup table. This field is qualified with the REQID field of the same entry when attempting to match. It is used to bitwise disable matches against PEXLaUDR[REQID] and a received packet's REQID.  Note: The reset value is 0xFFFF for entry 0 and 0x0000 for all other entries of the PEX lookup table.

## 28.5.1.6 PEX LUT Entry *a* Lower Data Register (PEXL0LDR)

### 28.5.1.6.1 Offset

Register	Offset
PEXL0LDR	804h

### 28.5.1.6.2 Function

The PEX LUT Entry *a* Lower Data Register (PEXL*a*LDR) represents the lower data fields of entry *a* within the PEX lookup table.

### 28.5.1.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EN	PL	BMT													
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W	Reserved		ICID													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.5.1.6.4 Fields

Field	Function
0 EN	Enable: The enable field of entry a within the PEX lookup table. This field indicates whether the entry participates in matching.  Note: The reset value is 0x1 for entry 0 and 0x0 for all other entries of the PEX lookup table.  0b - This entry does not participate in matching. 1b - This entry participates in matching.
1 PL	Priviledge Level: The priviledge level field of entry a within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.
2 BMT	Bypass memory translation: The bypass memory translation of entry a within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.
3-16 —	Reserved
17-31 ICID	Isolation context ID: The isolation context ID field of entry a within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.

### 28.5.1.7 PEX LUT Entry a Upper Data Register (PEXL1UDR - PEXL31UDR)

#### 28.5.1.7.1 Offset

For a = 1 to 31:

Register	Offset
PEXLaUDR	800h + (a × 8h)

### 28.5.1.7.2 Function

The PEX LUT Entry  $a$  Upper Data Register (PEXL $a$ UDR) represents the upper data fields of entry  $a$  within the PEX lookup table.

### 28.5.1.7.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									REQID							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									MASK							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.5.1.7.4 Fields

Field	Function
0-15 REQID	REQID: The Request ID of entry $a$ of the PEX lookup table. This field is qualified by the MASK field PEXLaUDR[MASK] of the same PEX lookup table entry when attempting to match.
16-31 MASK	MASK: The mask field of entry $a$ of the PEX lookup table. This field is qualified with the REQID field of the same entry when attempting to match. It is used to bitwise disable matches against PEXLaUDR[REQID] and a received packet's REQID.  Note: The reset value is 0xFFFF for entry 0 and 0x0000 for all other entries of the PEX lookup table.

## 28.5.1.8 PEX LUT Entry $a$ Lower Data Register (PEXL $1$ LDR - PEXL $31$ LDR)

### 28.5.1.8.1 Offset

For  $a = 1$  to 31:

Register	Offset
PEXL $a$ LDR	804h + ( $a \times 8h$ )

### 28.5.1.8.2 Function

The PEX LUT Entry  $a$  Lower Data Register (PEXL $a$ LDR) represents the lower data fields of entry  $a$  within the PEX lookup table.

### 28.5.1.8.3 Diagram

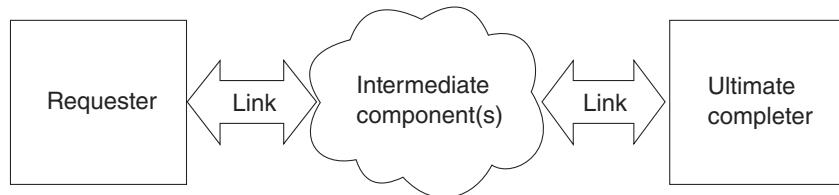
Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EN	PL	BMT						Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		ICID													
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 28.5.1.8.4 Fields

Field	Function
0 EN	Enable: The enable field of entry $a$ within the PEX lookup table. This field indicates whether the entry participates in matching.  Note: The reset value is 0x1 for entry 0 and 0x0 for all other entries of the PEX lookup table.  0b - This entry does not participate in matching. 1b - This entry participates in matching.
1 PL	Priviledge Level: The priviledge level field of entry $a$ within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.
2 BMT	Bypass memory translation: The bypass memory translation of entry $a$ within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.
3-16 —	Reserved
17-31 ICID	Isolation context ID: The isolation context ID field of entry $a$ within the PEX lookup table. If this entry matches, this field will be added to the outgoing AXI request sideband.

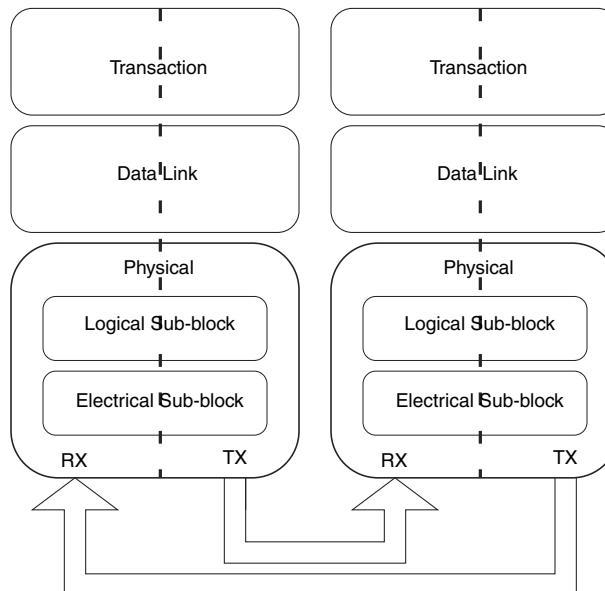
## 28.6 Functional Description

The PCI Express protocol relies on a requestor/completer relationship where one device requests that some desired action be performed by some target device and the target device completes the task and responds. Usually the requests and responses occur through a network of links, but to the requester and to the completer, the intermediate components are transparent.



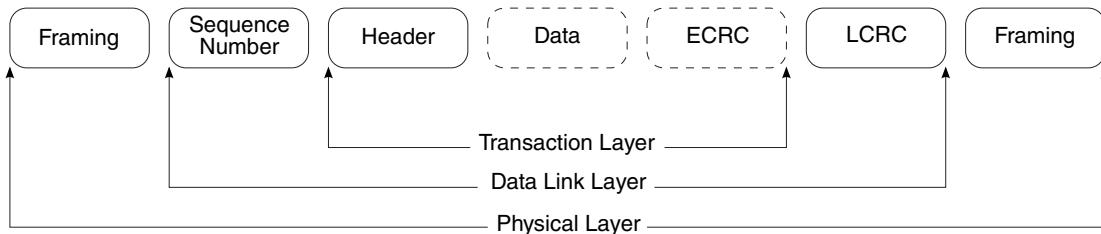
**Figure 28-3. Requester/Completer Relationship**

Each PCI Express device is divided into two halves—transmit (TX) and receive (RX), and each of these halves is further divided into three layers—transaction, data link, and physical, as shown in the following figure.



**Figure 28-4. PCI Express High-Level Layering**

Packets are formed in the transaction layer (TLPs) and data link layer (DLLPs), and each subsequent layer adds the necessary encodings and framing, as shown in the following figure. As packets are received, they are decoded and processed by the same layers but in reverse order, so they may be processed by the layer or by the device application software.

**Figure 28-5. PCI Express Packet Flow**

## 28.6.1 Architecture

This section describes implementation details of the PCI Express controller.

### 28.6.1.1 PCI Express Transactions

The following table contains the list of transactions that the PCI Express controller supports as an initiator and a target.

**Table 28-5. PCI Express Transactions**

PCI Express Transaction	Supported as an Initiator	Supported as a Target	Definition
Mrd	Yes	Yes	Memory Read Request
MRdLk	No	No	Memory Read Lock. As a target, CplLk with UR status is returned.
MWr	Yes	Yes	Memory Write Request to memory-mapped PCI-Express space
IOrd	Yes	No	I/O Read request. As a target, Cpl with UR status is returned.
IOWr	Yes	No	I/O Write Request. As a target, Cpl with UR status is returned.
CfgRd0	Yes	No	Configuration Read Type 0.
CfgWr0	Yes	No	Configuration Write Type 0.
CfgRd1	Yes	No	Configuration Read Type 1. As a target, Cpl with UR status is returned.
CfgWr1	Yes	No	Configuration Write Type 1. As a target, Cpl with UR status is returned.
Msg	Yes	Yes	Message Request. Message without data is not forwarded to memory.
MsgD	Yes	No	Message Request with Data payload.
Cpl	Yes	Yes	Completion without Data
CplD	Yes	Yes	Completion with Data
CplLk	No	Yes	Completion for Locked Memory Read without Data. The only time that CplLk is returned with UR status is when the controller receives a MRdLk command.
CplDLk	No	No	Completion for Locked Memory Read with Data

## 28.6.1.2 Lane Reversal

PCI Express lane reversal is supported as part of the PCI Express training operations where the lanes are reversed between the two chips at the end of a PCI Express link. This can occur because of chip packaging, pinning or board placement. Lane reversal is performed automatically as part of the lane-numbering negotiation after the link-width negotiation is complete.

The following table describes the supported configurations for the x4 controller.

**Table 28-6. x4 controller lane reversal**

Link configuration	Lane 0	Lane 1	Lane 2	Lane 3
x4 link without lane reversal	0	1	2	3
x2 link without lane reversal	0	1	—	—
x4 link with lane reversal	3	2	1	0
x2 link with lane reversal	1	0	—	—

**NOTE:** The numbers shown in this table (0-3) are the lane numbers assigned to each lane as a result of link initialization and configuration. — indicates that the lane is not part of the configured link.

### NOTE

The maximum number of lanes used in the lane reversal case cannot exceed the maximum number of lanes allocated for the PCI Express interface in use as determined by the SerDes options RCW settings (SRDS\_PRTCL\_Sn).

## 28.6.1.3 Transaction ordering rules

In general, transactions are serviced in the order that they are received.

## Functional Description

However, transactions can be reordered as they are sent due to a stalled condition such as a full internal buffer. The following table describes the transaction ordering rules for this device.

**Table 28-7. PCI Express controller TLP transaction ordering rules**

Row pass column?		Posted request	Non-posted request		Completion	
		Memory write or message request (col 2)	Read request (Col 3)	I/O or Configuration write request (col 4)	Read completion (Col 5)	I/O or configuration write completion (col 6)
Posted request	Memory write or message request (row A)	a) No <sup>1</sup> b) No <sup>1</sup>	Yes	Yes	a) Yes <sup>2</sup> b) N/A <sup>3</sup>	a) Yes <sup>2</sup> b) N/A <sup>3</sup>
Non-posted request	Read request (row B)	No	No	No	a) No <sup>4</sup> b) Yes <sup>5</sup>	a) No <sup>4</sup> b) Yes <sup>5</sup>
	I/O or configuration write request (Row C)	No	No	No	a) No <sup>4</sup> b) Yes <sup>5</sup>	a) No <sup>4</sup> b) Yes <sup>5</sup>
Completion	Read completion (row D)	a) No <sup>6</sup> b) Yes <sup>7</sup>	Yes	Yes	a) No <sup>8</sup> b) No <sup>8</sup>	No
	I/O or configuration write completion (row E)	a) No <sup>6</sup> b) Yes <sup>7</sup>	Yes	Yes	No	No

1. Regardless of the setting of the relaxed ordering (RO) bit, a posted request cannot bypass another posted request.
2. Regardless of the setting of the relaxed ordering bit, a posted request can always bypass a completion.
3. N/A indicates that the original rules at these entries defined by the *PCI Express Base Specification* do not apply.
4. A non-posted request cannot bypass a completion if the relaxed ordering bit is cleared (that is, RO = 0).
5. A non-posted request can bypass a completion if the relaxed ordering bit is set (that is, RO = 1).
6. A read completion, I/O write completion, or configuration write completion cannot bypass a posted request if the relaxed ordering bit is cleared (that is, RO = 0).
7. A read completion, I/O write completion, or configuration write completion can bypass a posted request if the relaxed ordering bit is set (that is, RO = 1).
8. Regardless of the setting of the relaxed ordering bit, a read completion cannot bypass another read completion.

In general, the following points summarize the ordering rules for sending the next outstanding request:

- A posted request can bypass all other transactions except another posted request.
- A non-posted request cannot bypass posted or other non-posted requests, but it can bypass a completion if the relaxed ordering (RO) bit is set.(See [Table 28-7](#)).
- A completion can bypass posted requests if the relaxed ordering (RO) bit is set and can bypass non-posted transactions. However, a completion cannot bypass other completions.

## 28.6.1.4 Internal Address Translation Unit

The core uses the iATU to replace the TLP address and TLP header fields in the current TLP request header. This section presents the following topics:

- [iATU Overview](#)
- [Programming the iATU](#)
- [Outbound iATU Operation](#)
- [Inbound iATU Operation](#)

### 28.6.1.4.1 iATU Overview

Address translation is used for mapping different address ranges to different memory spaces supported by your application. A typical example maps the internal platform memory space to PCI memory space. The iATU supports type translation. Without address translation, your application address is passed to/from the PCIe TLPs directly through the internal platform . You can program the iATU to implement your own address translation scheme without the need for additional external hardware.

#### Inbound Features:

- Match mode operation for MEM TLPs. No translation for completions.

#### NOTE

Address match mode must be used.

- Programmable TLP header field matching.
  - TYPE
- 6 address regions programmable for location and size.
- Programmable enable/disable per region.
- Automatic FMT field translation between three DWORDs and four DWORDs for 64-bit addresses.
- ECAM Configuration Shift mode to allow a 256 MB CFG1 space to be located anywhere in the 64-bit address space.
- Supports regions from 4 kB to 4 GB in size.

#### Outbound Features:

- Address Match mode operation for MEM and I/O, CFG, and MSG TLPs. No translation for completions.
- Supports type translation through TLP type header field replacement for MEM or I/O types to MSG/CFG types.

- Includes posted to non-posted translation (for example, MWr to CfgWr0)
- No translation from completions
- Programmable TLP header field replacement.
  - TYPE/TD/TC/ATTR/MSG-Code
- 6 address regions programmable for location and size.
- Programmable enable/disable per region.
- Automatic FMT field translation between three DWORDs and four DWORDs for 64-bit addresses.
- Configuration Shift mode. Optimizes the memory footprint of CFG accesses destined for the internal platform interface in a multifunction device.
- Response code which defines the completion status to return for accesses matching a region.
- Supports regions from 4 kB to 4 GB in size.

### NOTE

The default behavior of the ATU when there is no address match in the outbound direction or no TLP attribute match in the inbound direction, is to pass the transaction through.

#### 28.6.1.4.2 Programming the iATU

You can access the iATU registers through the DBI interface or through PCIe CFG accesses. The following registers are used for programming the iATU.

**Table 28-8. iATU Register Map**

Byte Offset	Description
0x900	iATU Index Register
0x904	iATU Region Control 1 Register
0x908	iATU Region Control 2 Register
0x90C	iATU Region Lower Base Address Register
0x910	iATU Region Upper Base Address Register
0x914	iATU Region Limit Address Register
0x918	iATU Region Lower Target Address Register
0x91C	iATU Region Upper Target Address Register
0x920	iATU Region Control 3 Register

The iATU registers are programmed through an indirect addressing scheme (using an index register) to reduce the address footprint in the PCI Express extended configuration space. The index register has a “Region Direction” bit to determine whether an inbound or outbound region is being accessed and a “Region Index” field to determine which region to program/read when accessing the other address translation registers.

### 28.6.1.4.3 Outbound iATU Operation

This section describes the processing of outbound requests by the iATU. This section presents the following topics:

- Overview (Address Match Mode)
- RID BDF Number Replacement
- CFG Handling
- CFG Shift Feature
- FMT Translation
- No Address Match Result
- Writing to a MRdLk Region
- Outbound Programming Example

#### 28.6.1.4.3.1 Overview (Address Match Mode)

The address field of each request MEM and I/O TLP is checked to see if it falls into any of the enabled address regions defined by the “Start” and “End” addresses as defined in Figure 28-6.

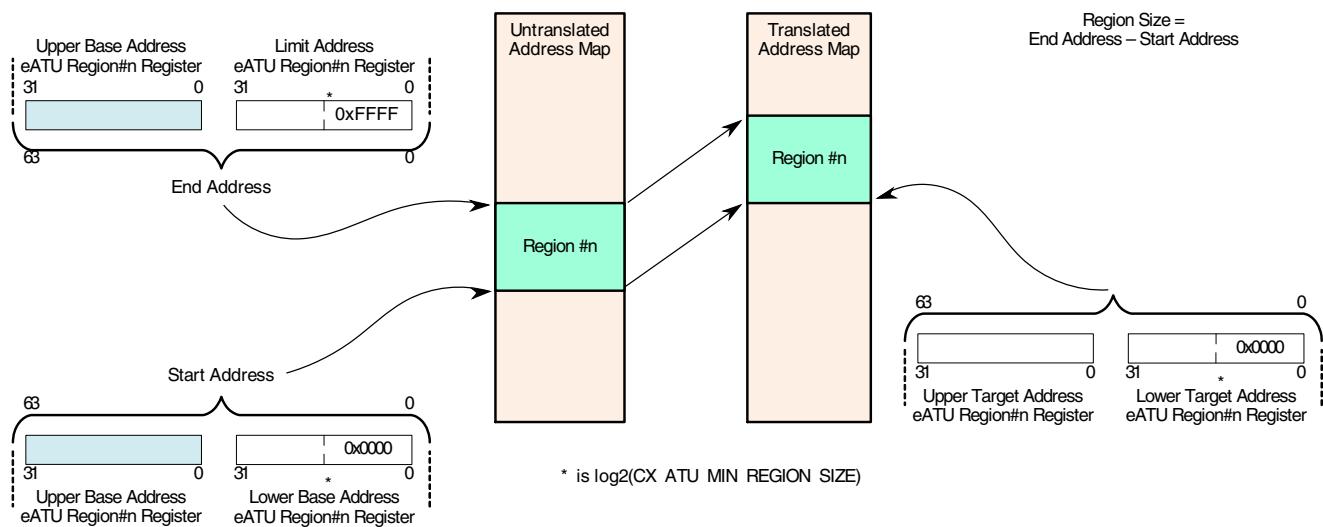
#### **NOTE**

When the “Region Enable” bit of the Region Control 2 register is “0”, then that region is not used for address matching.

When an address match is found, then the TLP address field is modified as follows:

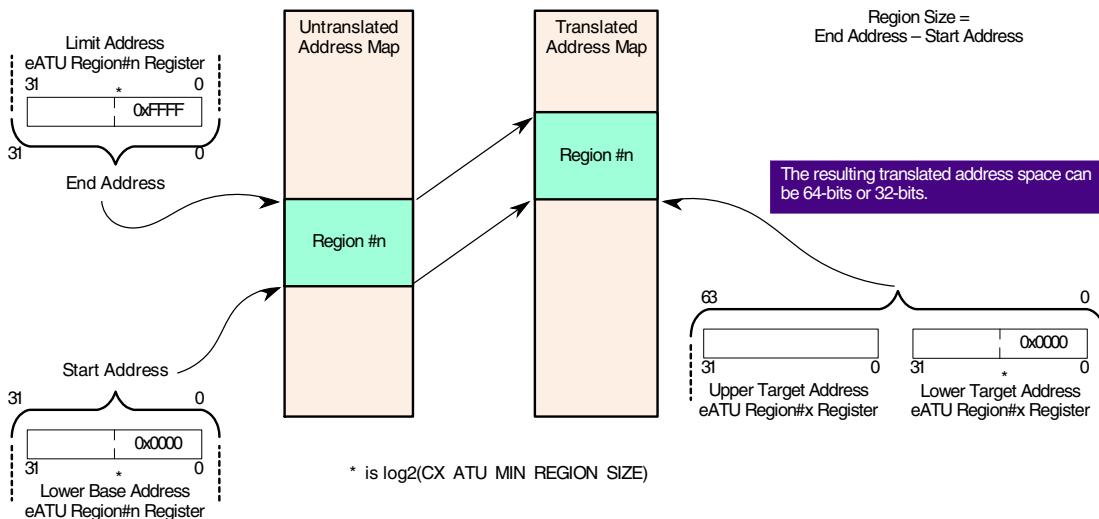
Address = Address - Base Address + Target Address and the TYPE, TD, TC, and ATTR header fields are replaced with the corresponding fields in the “iATU Control 1 Register” (IATU\_REGION\_CTRL\_1\_OFF\_OUTBOUND\_0). When your application internal platform address field matches more than one of the 6 address regions, then the first (lowest of the numbers from 0 to 5) enabled region to be matched is used. See [No Address Match Result](#) for details on what happens when there is no address match. This operational mode (called “Address Match Mode”) is always used for outbound translation.

## Functional Description



**Figure 28-6. iATU Address Region Mapping: Outbound and Inbound (Address Match Mode), 64-bit Address**

When the PCIe core is operating with 32-bit addresses, then the operation is defined as in figure below.



**Figure 28-7. iATU Address Region Mapping: Outbound and Inbound (Address Match Mode), 32-bit Address**

The upper 32 bits of the Target Address register always forms the upper 32 bits of the translated address because:

- The maximum region size is 4 GB.
- A region must not cross a 4 GB boundary.

In the figure, CX\_ATU\_MIN\_REGION\_SIZE specifies the minimum size of an address translation region and is 4 KB.

#### 28.6.1.4.3.2 RID BDF Number Replacement

When there is a successful address match on an outbound TLP, then the function number used in generating the function part of the requester ID field of the TLP is taken from the 3-bit “Function Number” field of the “iATU Control 1 Register” (IATU\_REGION\_CTRL\_1\_OFF\_OUTBOUND\_0). The value in this field must be 0x0 unless multifunction operation is enabled (the number of PFs is greater than 1).

#### NOTE

The requester ID uses the 8:5:3 bit PCI Bus.Device.Function (BDF) format.

#### 28.6.1.4.3.3 CFG Handling

The iATU supports translation of I/O and MEM TLPs to CFG TLPs. The 16-bit Bus.Device.Function (BDF) is derived from bits [31:16] of the “iATU Lower Target Address Register” (IATU\_LWR\_TARGET\_ADDR\_OFF\_OUTBOUND\_0). You can shift/remap the BDF within the PCIe address space using the [CFG Shift Feature](#). Address translation of CFG TLPs is also possible with the iATU.

#### 28.6.1.4.3.4 CFG Shift Feature

This expander feature can be enabled by setting the “CFG Shift” bit of the “iATU Control 2 Register” (IATU\_REGION\_CTRL\_2\_OFF\_OUTBOUND\_0). This shifts/maps the BDF, which is bits [27:12] of the target address, up to bits [31:16] of the translated address. This allows all outgoing I/O and MEM TLPs (that have been translated to CFG) to be mapped into any 256 MB region of the PCIe address space. This scheme also supports the Enhanced Configuration Address Mapping (ECAM) mechanism from Section 7.2.2 of the PCI Express Base 3.0 Specification, revision 1.0. ECAM supports the mapping (MEM to CFG translation) from memory address space to PCIe configuration space address as defined in [Table 28-9](#). ECAM maps bits [27:12] of the untranslated MEM TLP to become the BDF of the resulting CFG TLP.

**Table 28-9. ECAM Mapping**

Memory Address Bits	PCIe Configuration Space
27:20	Bus Number
19:15	Device Number
14:12	Function Number
11:8	Extended Register Number
7:2	Register Number

#### 28.6.1.4.3.5 FMT Translation

The iATU automatically sets the TLP format field for three DWORDs when it detects all zeroes in the upper 32 bits of the translated address. Otherwise, it sets it to four DWORDs when it detects a 64-bit address (that is, when there is a “1” in the upper 32 bits of the translated address). When the original address and the translated address are of different format, the iATU ensures that the TLP header size matches the translated address format.

#### 28.6.1.4.3.6 No Address Match Result

When there is no address match then the address is untranslated but the TLP header information comes from the relevant fields on the internal platform interface.

#### 28.6.1.4.3.7 Writing to a MRdLk Region

When there is a successful address match for an outbound write and the type header field in the “iATU Control 1 Register” is “00001” indicating a locked MEM transfer, then the core sets the type field to “0000” (MEM).

#### 28.6.1.4.3.8 Outbound Programming Example

Define outbound region 1 as a 64 kB I/O region from 0x8000\_0000\_D000\_0000 to 0x8000\_0000\_D000\_FFFF, to be mapped to 0x00010000 in the PCIe I/O space.

1. Setup the Index Register.

Write 0x00000001 to Address { 0x700 + 0x200 } to set outbound region 1 as the current region

2. Setup the Region Base and Limit Address Registers.

Write 0xd0000000 to Address {0x700 +0x20C} to set the Lower Base Address.

Write 0x80000000 to Address {0x700 +0x210} to set the Upper Base Address.

Write 0xd000ffff to Address {0x700 +0x214} to set the Limit Address.

3. Setup the Target Address Registers.

Write 0x00010000 to Address {0x700 +0x218} to set the Lower Target Address.

Write 0x00000000 to Address {0x700 +0x21C} to set the Upper Target Address.

4. Configure the region through the Region Control 1 Register.

Write 0x00000002 to Address {0x700 +0x204} to define the type of the region to be I/O.

##### 5. Enable the region.

Write 0x80000000 to Address {0x700 +0x208} to enable the region.

#### 28.6.1.4.4 Inbound iATU Operation

This section discusses how the iATU processes inbound requests. The topics are:

- [Overview](#)
- [MEM Match Modes](#)
- [CFG Handling \(Upstream Port\)](#)
- [FMT Translation](#)
- [Inbound Programming Example](#)

##### 28.6.1.4.4.1 Overview

###### **NOTE**

- The main difference between inbound and outbound iATU operation is that the TLP type is never changed in the inbound direction. Instead, the type field is used for more precise matching. Other fields can also be optionally used to further refine the matching process.
- 

The following translation rules and limitations apply:

- When there is no match, then the address is untranslated. In addition
- TLPs destined for the configuration registers in an upstream port are not translated.
- TLPs that are not error-free (ECRC, malformed and so on) are not translated.
- Address translation of all TLP types (MEM, CFG, and MSG) except completion is supported in Address Match mode.

The setting of the “Match Mode” field in the “iATU Control 2 Register” (IATU\_REGION\_CTRL\_2\_OFF\_OUTBOUND\_0) determines how iATU inbound matching is done for each TLP type.

**Table 28-10. Determination of Match Mode By TLP Type and “Match Mode” Field of iATU Control 2 Register**

	Match Mode = 0	Match Mode = 1
MEM	Address Match Mode	BAR Match Mode
CFG0	Routing ID Match Mode	Accept Mode
MSG/MSGD	Address Match Mode	Vendor ID Match Mode

#### 28.6.1.4.4.2 MEM Match Modes

Inbound address translation for MEM TLPs operates in one of two matching modes as determined by the “Inbound Match Mode” field in the “iATU Control 2 Register”.

##### 28.6.1.4.4.2.1 Address Match Mode

The operation is similar to [Outbound iATU Operation](#). The address field of each request TLP is checked to see if it falls into any of the enabled address regions as shown in [Figure 28-6](#).

##### NOTE

When the “Region Enable” bit of the “Region Control 2 Register” is “0”, then that region is not used for address matching.

##### NOTE

The target (translation) address and base address of the inbound iATU window must be aligned based on the window size programmed in the corresponding BAR MASK register.

When an address match is found then the TLP address field is modified as follows:

$$\text{Address} = \text{Address} - \text{Base Address} + \text{Target Address}$$

When the TLP address field matches more than one of the 6 address regions, then the first (lowest of the numbers from 0 to 5) enabled region to be matched is used.

##### 28.6.1.4.4.2.2 BAR Match Mode

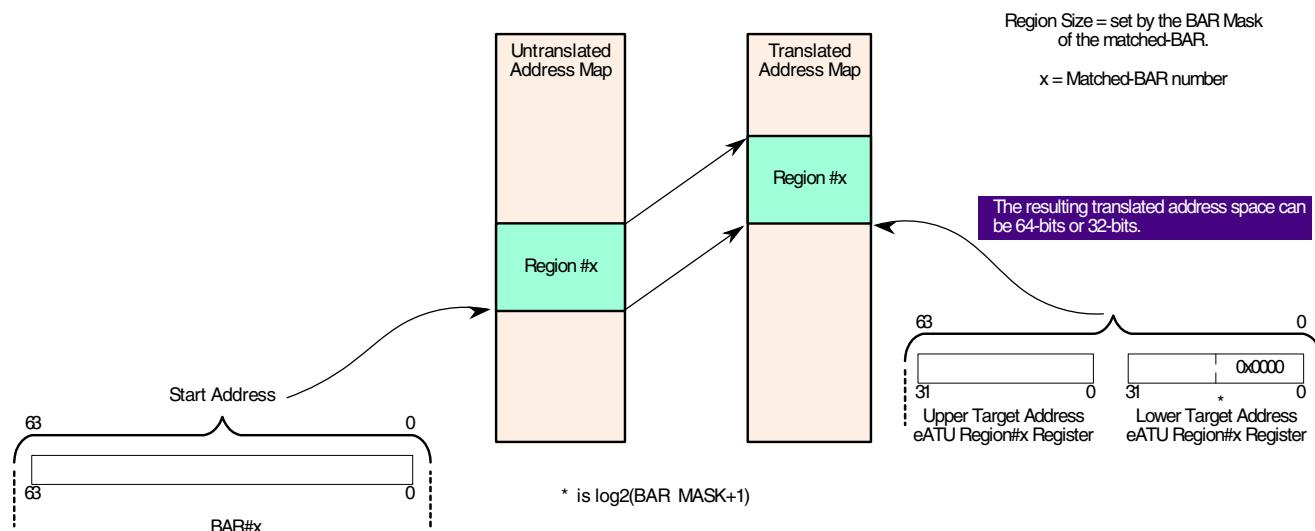
Looking for an address match is a two-step process.

1. The standard internal PCI Express BAR Matching Mechanism checks if the address field of any MEM request TLP falls into any address region defined by the enabled BAR addresses and masks.
2. When a matched BAR is found, then the iATU compares the BAR ID to the “BAR Number” field in the “iATU Control 2 Register” for all enabled regions. [Figure 28-8](#) and [Figure 28-9](#) provide more details on inbound translation in BAR Match Mode.

### NOTE

BAR Match Mode can only be used for MEM transactions

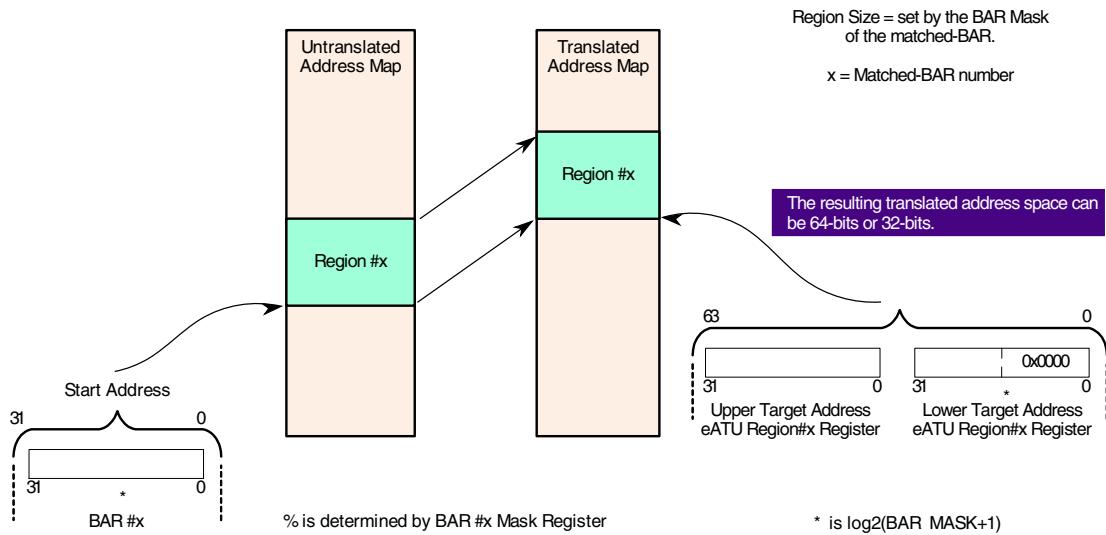
When the PCIe core is operating with 64-bit BARs, the operation is defined as in figure below.



**Figure 28-8. iATU Address Region Mapping: Inbound (BAR Match Mode), 64-bit BAR**

In address match mode, when the address range does not match one of its BAR ranges in an upstream port, then the device rejects the request with unsupported request (UR) completion status and no translation occurs.

When the PCIe core is operating with 32-bit BARs, the operation is defined as in figure below.



**Figure 28-9. iATU Address Region Mapping: Inbound (BAR Match Mode), 32-bit BAR**

#### 28.6.1.4.4.3 CFG Handling (Upstream Port)

The PEX controller normally routes CFG TLPs to the configuration registers without translating them. The iATU only translates CFG0 TLPs that the PEX controller has routed to the internal platform. Inbound address translation for CFG0 TLPs operates in one of two matching modes as determined by the “Inbound CFG0 Match Mode” field in the “iATU Control 2 Register”.

**Routing ID Match Mode:** The operation is similar to [Outbound iATU Operation](#). The routing ID of the inbound CFG0 TLP must fall within the Base and Limit of the defined iATU region for matching to proceed. The iATU interprets the routing ID (Bytes 8-11 of TLP header) as an address. This corresponds to the upper 16 bits of the address in MEM and I/O transactions.

**Accept Mode:** The PEX controller always accepts CFG0 TLPs even when the CFG bus number does not match the current bus number of the device. This mode follows that behavior. The routing ID of received CFG0 TLPs are ignored when determining a match.

**CFG1 Transactions:** For CfgRd1/CfgWr1 transactions, the base and limit addresses could enclose the entire 32-bit 4 GB memory space with the routing ID forming the upper 16 bits. The target address maps these CFG transactions to anywhere in application address space.

#### 28.6.1.4.4.4 FMT Translation

The iATU automatically sets the TLP format field for three DWORDs when it detects all zeroes in the upper 32 bits of the translated address. Otherwise it sets it to four DWORDs when it detects a 64-bit address (when there is a “1” in the upper 32 bits of the translated address). When the original address and the translated address are of a different format then the iATU ensures that the TLP header size matches the translated address format.

#### 28.6.1.4.4.5 Inbound Programming Example

##### 28.6.1.4.4.5.1 *Address match mode*

Define inbound Region 0 as: MEM region matching TLPs with addresses in the range 0x00010000 to 0x0005ffff mapped to 0x1000\_0000\_2000\_0000 - 0x1000\_0000\_2004\_ffff in your application memory space

1. Setup the Index Register.

Write 0x80000000 to Address { 0x700 + 0x200 } to set inbound region 0 as the current region

2. Setup the Region Base and Limit Address Registers.

Write 0x00010000 to Address {0x700 + 0x20C} to set the Lower Base Address.

Write 0x00000000 to Address {0x700 + 0x210} to set the Upper Base Address.

Write 0x0005ffff to Address {0x700 + 0x214} to set the Limit Address

3. Setup the Target Address Registers.

Write 0x20000000 to Address {0x700 + 0x218} to set the Lower Target Address.

Write 0x10000000 to Address {0x700 + 0x21C} to set the Upper Target Address.

4. Configure the region through the Region Control 1 Register.

Write 0x00000000 to Address {0x700 + 0x204} to define the type of the region to be MEM.

5. Enable the region.

Write 0x80000000 to Address {0x700 + 0x208} to enable the region in address match mode.

**NOTE**

Defined MEM regions must either match a BAR or be outside of the base and limit ranges defined for the port in the Type 1 configuration header.

### 28.6.1.5 Memory Space Addressing

A PCI Express memory transaction can address a 32- or 64-bit memory space. As an initiator, the controller is capable of sending 32- or 64-bit memory packets. Any transaction from the internal platform that (after passing through the translation mechanism) has a translated address greater than 4G is sent as a 64-bit memory packet. Otherwise, a 32-bit memory packet is sent. As a target device, the controller is capable of decoding 32- or 64-bit memory packets. This is done through two 32-bit inbound windows and two 64-bit inbound windows. All inbound addresses are translated to 40-bit internal platform addresses.

**Table 28-11. PCI Express Memory Transactions**

Transaction	Type[4:0]	FMT[1]	FMT[0]
MRd (32-bit address)	00000	0	0
MRd (64-bit address)	00000	0	1
MWr (32-bit address)	00000	1	0
MWr (64-bit address)	00000	1	1

### 28.6.1.6 I/O Space Addressing

As an initiator, the controller can send I/O transactions in RC mode only. This can be done by programming one of the outbound translation window's attribute to send I/O transactions. The controller does not support I/O transactions as a target.

**Table 28-12. PCI Express I/O Transactions**

Transaction	Type	FMT[1]
IORD	00010	0
IOWR	00010	1

The PCI Express Base Specification restricts I/O space to 4 GB (32-bit I/O address).

## 28.6.1.7 Configuration Space Addressing

The controller can generate outbound configuration transactions in RC mode only. The controller does not support inbound configuration transactions in RC mode.

**Table 28-13. PCI Express Configuration Transactions**

Transaction	Type	FMT[1]
CfgRd0	00100	0
CfgWr0	00100	1
CfgRd1	00101	0
CfgWr1	00101	1

Note that all configuration transactions sent on PCI Express require a response, regardless whether they are read or a write configuration transactions.

## 28.6.1.8 Messages

This section describes outbound message generation and inbound message reception.

### 28.6.1.8.1 Outbound Message Generation

Software can generate the following outbound message transactions:

Outbound Message	Generated by
PME_Turn_Off	Writing to SCFG_PEXPMECR. See <a href="#">PCI Express PM turnoff message support</a>
Set_Slot_Power_Limit	Writing to the Slot Power Limit Scale and Slot Power Limit Value fields of the Slot Capabilities Register in the PCI Express Capabilities Structure.

### 28.6.1.8.2 Inbound Messages

The following table provides a complete list of supported inbound messages in RC mode.

**Table 28-14. PCI Express RC Inbound Message Handling**

Name	Code[7:0]	Routing[2:0]	Action
<b>INTx Interrupt Signaling</b>			
Assert_INTA	0010 0000	100	Assert INTA virtual wire. Sent to GIC.
Assert_INTB	0010 0001	100	Assert INTB virtual wire. Sent to GIC.
Assert_INTC	0010 0010	100	Assert INTC virtual wire. Sent to GIC.

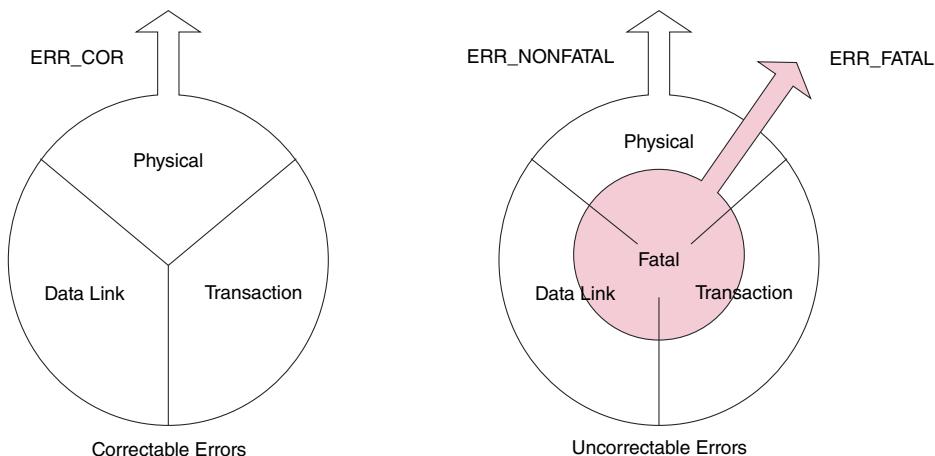
*Table continues on the next page...*

**Table 28-14. PCI Express RC Inbound Message Handling (continued)**

Name	Code[7:0]	Routing[2:0]	Action
Assert_INTD	0010 0011	100	Assert INTD virtual wire. Sent to GIC.
Deassert_INTA	0010 0100	100	De-assert INTA virtual wire. Sent to GIC.
Deassert_INTB	0010 0101	100	De-assert INTB virtual wire. Sent to GIC.
Deassert_INTC	0010 0110	100	De-assert INTC virtual wire. Sent to GIC.
Deassert_INTD	0010 0111	100	De-assert INTD virtual wire. Sent to GIC.
<b>Power Management</b>			
PM_PME	0001 1000	000	Generate interrupt to GIC
PME_TO_Ack	0001 1011	101	Generate interrupt to GIC
<b>Error Signaling</b>			
ERR_COR	0011 0000	000	Generate interrupt to GIC
ERR_NONFATAL	0011 0001	000	Generate interrupt to GIC
ERR_FATAL	0011 0011	000	Generate interrupt to GIC

### 28.6.1.9 Error Handling

The PCI Express specification classifies errors as correctable and uncorrectable. Correctable errors result in degraded performance, but uncorrectable errors generally result in functional failures. As shown in the figure below, uncorrectable errors can further be classified as fatal or non-fatal.

**Figure 28-10. PCI Express Error Classification**

### 28.6.1.9.1 PCI Express Error Logging and Signaling

The figure below shows the PCI Express-defined sequence of operations related to signaling and logging of errors detected by a device. Note that the PCI Express controller on this device supports the advanced error handling capabilities shown within the dotted lines.

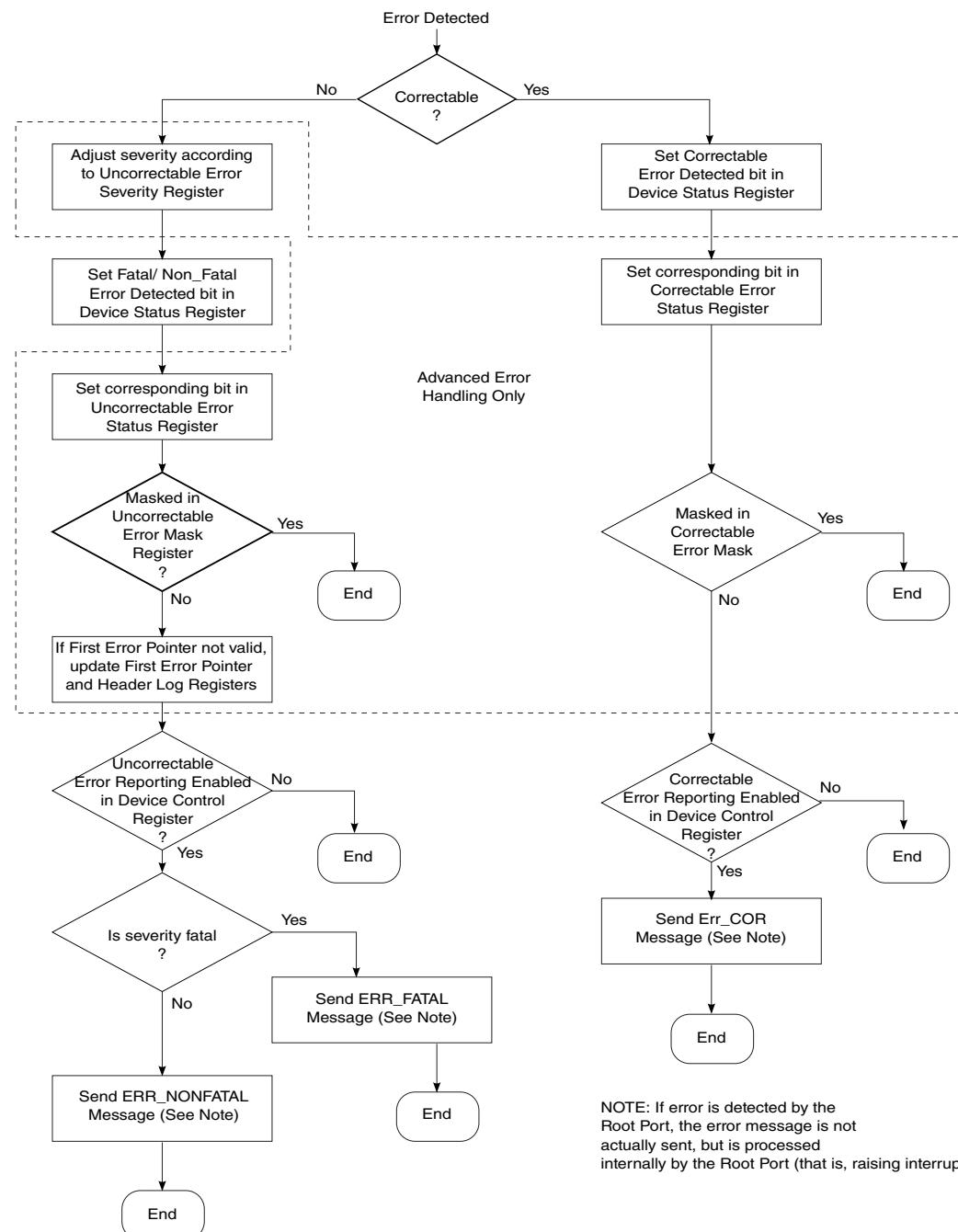


Figure 28-11. PCI Express Device Error Signaling Flowchart

## 28.6.2 Interrupts

Both message signaled interrupts (MSI) and legacy INTx are supported.

- MSI interrupts are handled by the SCFG module. See [PCI Express MSI implementation](#) for more information.
- Legacy INTx interrupts are handled by virtual-wire message transactions. See [Inbound Messages](#).

## 28.6.3 Initial Credit Advertisement

To prevent overflowing of the receiver's buffers and for ordering compliance purposes, the transmitter cannot send transactions unless it has enough flow control (FC) credits to send. Each device maintains an FC credit pool. The FC information is conveyed between the two link partners by DLLPs during link training (initial credit advertisement). The transaction layer performs the FC accounting functions. One FC unit is four DWs (16-bytes) of data.

**Table 28-15. Initial credit advertisement**

Credit Type	Initial Credit Advertisement
PH (Memory Write, Message Write)	8
PD (Memory Write, Message Write)	128
NPH (Memory Read, IO Read, Cfg Read, Cfg Write)	12
NPD (IO Write, Cfg Write)	2
CPLH (Memory Read Completion, IO R/W Completion, Cfg R/W Completion)	Infinite
CPLD (Memory Read Completion, IO Read Completion, Cfg Read Completion)	Infinite

## 28.6.4 Power Management

All device power states are supported with the exception of D3cold. Only L0s ASPM mode is supported if enabled by configuring the Link Control register's bits 1-0 in configuration space. Note that there is no power saving in the controller when the device is put into a non-D0 state. The only power saving is the I/O drivers when the controller is put into a non-L0 link state.

**Table 28-16. Power Management State Supported**

Component D-State	Permissible Interconnect State	Action
D0	L0, L0s	In full operation.
D1	L0, L0s, L1	All outbound traffic is stalled. All inbound traffic is thrown away. The only exceptions are PME messages and configuration transactions.
D2	L0, L0s, L1	All outbound traffic is stalled. All inbound traffic is thrown away. The only exceptions are PME messages and configuration transactions.
D3hot	L0, L0s, L1, L2/L3 Ready	All outbound traffic is stalled. All inbound traffic is thrown away. The only exceptions are PME messages and configuration transactions. Note that if a transition of D3hot to D0 occurs, a reset is performed to the controller's configuration space. In addition, link training restarts.
D3cold	L3	Completely off.

#### 28.6.4.1 L2/L3 Ready Link State

The L2/L3 Ready link state is entered after the EP device is put into a D3hot state followed by a PME\_Turn\_Off/PME\_TO\_Ack message handshake protocol. Exiting this state requires a POR reset or a WAKE\_B signal from the EP device.

In RC mode, the WAKE\_B signal from the EP device can be connected to one of the external interrupt inputs to service the WAKE\_B request.

#### 28.6.5 Hot Reset

When a hot reset condition occurs, the controller initiates a clean-up of all outstanding transactions and returns to an idle state. All configuration register bits that are non-sticky are reset. Link training takes place subsequently. The device is permitted to generate a hot reset condition on the bus when it is configured as an RC device by setting the "Secondary Bus Reset" bit in the Bridge Control Register in the configuration space.

### 28.7 Initialization/Application Information

#### 28.7.1 Configuring the chip for inbound CCSR accesses

The chip must be configured using the following procedure prior to receiving any inbound memory transactions targeting CCSR space:

1. Program COHERENCY\_CONTROL\_3\_OFF (offset 8E8h) = 0000\_0000h
2. Program COHERENCY\_CONTROL\_2\_OFF (offset 8E4h) = 0000\_0000h
3. Program COHERENCY\_CONTROL\_1\_OFF (offset 8E0h) = 1000\_0000h

This procedure establishes a boundary between CCSR and memory at 1000\_0000h in the system memory map.

**NOTE**

Any inbound memory transactions targeting CCSR space must set NO\_SNOOP = 1 in the TLP.

## **28.7.2 Poisoned TLP handling**

In some cases, the handling of inbound Poisoned TLPs needs special attention. Inbound Poisoned TLP requests are dropped and the Poisoned TLP flag in the Uncorrectable Error Status Register (bit 12) is set, provided it is not masked. However, for an inbound Poisoned TLP response to an outbound request, the response is forwarded to the application as a completion and the Poisoned TLP flag in the Uncorrectable Error Status Register (bit 12) is set (again, provided it is not masked). This allows the requesting transaction to terminate without stalling, although the error is flagged. In this case, the error handler should cause the response to be discarded.

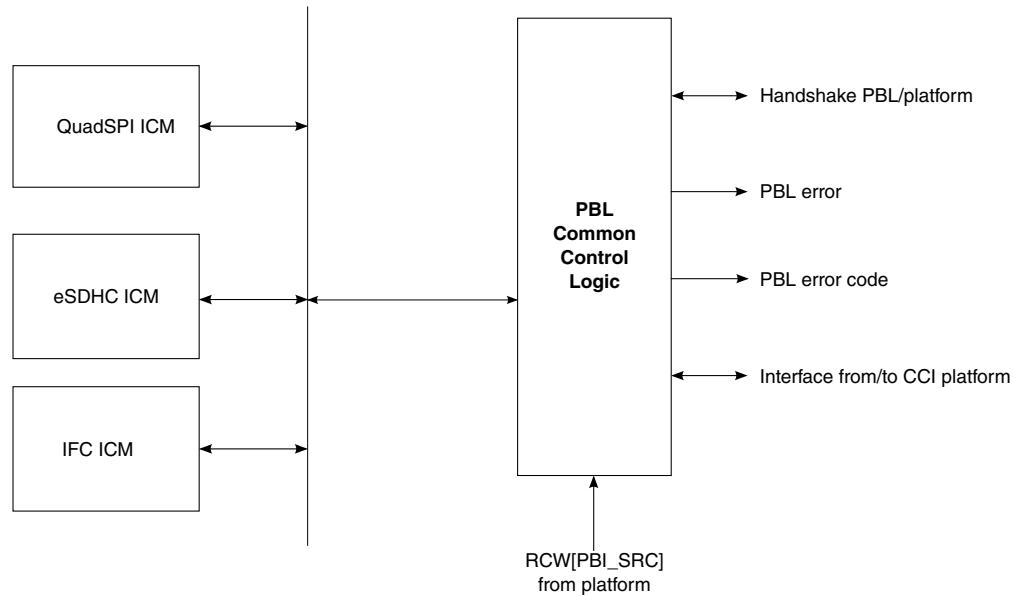
# Chapter 29

## Pre-Boot Loader (PBL)

### 29.1 Overview

The pre-boot loader (PBL) performs configuration register reads and writes to initialize the IFC FCM (NAND flash), eSDHC, QuadSPI interface, loads the RCW and pre-boot initialization commands from external memory device through IFC FCM (NOR flash), eSDHC, QuadSPI (QSPI\_CS\_A0) , and writes data to configuration registers or memory before the local cores are permitted to boot.

The following figure shows a block diagram of PBL, showing the functional organization of the module, which includes interface command modules (ICMs) for eSDHC, QuadSPI, IFC, and a common control module.



**Figure 29-1. Pre-boot loader (PBL) block diagram**

## 29.2 Features summary

PBL supports the following features:

- Initializes and loads RCW/pre-boot initialization commands from eSDHC SD/eMMC interface
  - Supports eMMC
    - 1 bit data transfer on eMMC
    - Compliant with version 4.2 of the specification
    - Standard capacity (< 2 GB)
    - High-capacity (> 2 GB)
  - Supports Secure Digital (SD) memory card
    - 1 bit data transfer on SD interface
    - Compliant with version 2.00 of the specification
    - Standard capacity (< 2 GB)
    - High-capacity (> 2 GB)
- Initializes and loads RCW/pre-boot initialization commands from IFC FCM NAND/NOR interface
- Initializes and loads RCW/pre-boot initialization commands from QuadSPI interface
- Reports error upon receiving error response from each interface
- Reports error if RCW byte count is not 64 bytes
- Reports error if RCW address offset is not the specific 64 bytes aligned offset to RCW status registers (RCWSR $n$ ) to which RCW contents are written
- Reports error in secure boot mode if pre-boot initialization address offset is inside of protected CCSR spaces
- Reports byte count and address misalignment error during pre-boot initialization
- Reports error if PBL command is in invalid format
- Reports error if there is no response from platform when timeout counter expires

## 29.3 Modes of operation

PBL operates in the following mode:

- Load both RCW and pre-boot initialization commands from individual interfaceQuadSPI flash memory.

## 29.4 Functional description

This section describes the following topics:

- Device configuration using reset configuration word (RCW)
- Device initialization by PBL
- Required format of data structure used by PBL
- RCW loading by PBL
- Pre-boot initialization command loading by PBL
- Reserved address space used as internal PBL commands
- Error codes

### 29.4.1 Device configuration using reset configuration word (RCW)

The reset configuration word (RCW) data contains reset configuration information that is loaded by the PBL from a memory device during reset.

The size of the RCW is 64 bytes (512 bits). All of the data read from the RCW source is written to the RCW status registers (RCWSR $n$ ) in the device configuration and pin control block. See DCFG\_RCWSR for more information. The RCWSRs have specific 64 bytes aligned address offsets that the PBL should ensure is written. Any address offset that is not that specific 64 bytes aligned address offset is reported by the PBL as an error to the platform, which asserts the RESET\_REQ\_B output pin upon receiving the error from the PBL. Refer the Reset configuration word (RCW) source section in Reset, Clocking, and Initialization chapter to see how each encoding for cfg\_rcw\_src selects a given source of RCW data.

### 29.4.2 Device initialization by PBL

The cfg\_rcw\_src configuration input determines the interface to use for retrieving the RCW data. Similarly, RCW[PBI\_SRC] determines the interface to use for pre-boot initialization. In these cases, the PBL initializes the QuadSPI, eSDHC, or IFC FCM (NAND flash) interface before accessing the RCW or PBI commands.

Note that IFC FCM (NOR flash) does not require initialization by the PBL even though it can be selected as the source for the RCW or PBI commands. Each interface command module (ICM) inside the PBL issues CCSR space accesses to setup each interface properly. If there is an error response during initialization, the PBL reports the error to the platform.

### **NOTE**

Only one interface can be used as the source of RCW and pre-boot initialization data. The PBL data structure must not be split across two interfaces.

#### **29.4.2.1 CCSR registers blocked from PBL during secure boot**

The following table lists the CCSR address ranges that are blocked from PBL during secure boot.

Address regions blocked independent of OSPR[PGE]

The table below specifies the address region blocked for write/read operation when OSPR[ITS]=1 or RCW[SB\_EN]=1, during PBI process. This blocking behavior is independent of OSPR[PGE].

**Table 29-1. CCSR registers blocked from PBL during secure boot**

Address range	Size	Description
0x150_0000-0x150_FFFF	64 KB	TZASC
0x151_0000-0x151_FFFF	64 KB	CSU
0x152_0000-0x152_FFFF	64 KB	Platform control
0x1E8_0000-0x1E8_FFFF	64 KB	Security fuse processor (SFP)
0x1E9_0000-0x1E9_FFFF	64 KB	SecMon
0x170_0000-0x17F_FFFF	1024 KB	SEC
0x900_0000	16MB	SMMU
0x2B0_0000	64 KB	Sys Counter
0x2AD_0000	64 KB	WDOG1
0x2AE_0000	64 KB	WDOG2
0x2A7_0000	64 KB	WDOG3
0x2A8_0000	64 KB	WDOG4
0x2A9_0000	64 KB	WDOG5

Address regions blocked when OSPR[PGE]=1

The table below specifies the address region blocked for read/write operation when OSPR[ITS]=1 or RCW[SB\_EN]=1 only when OSPR[PGE]=1, during PBI process.

**Table 29-2. Registers blocked from PBL during secure boot**

CCSR Offset	Size	Description
0x1EE_0000	2 KB	DCFG (SCRATCHWR offset at 0x1EE_0200)
0x1EE_1000	4 KB	Clocking (all registers)
0x1EE_2000	4 KB	RCPM (all registers)
0x157_0000	64 KB	SCFG (USB2 ICID, USB3 ICID, qDMA ICID, SATAICID, USB1 ICID, QE ICID, SDHC ICID, eDMA ICID, and ETR Stream ID)

### 29.4.3 Required format of data structure used by PBL

The RCW and pre-boot initialization commands share the same data structure format, regardless of the type of external memory device used as the source.

**Table 29-3. Required format of data structure used by PBL**

	0	1	2	3	4	5	6	7						
Preamble (required)	1	0	1	0	1	0	1	0						
	0	1	0	1	0	1	0	1						
	1	0	1	0	1	0	1	0						
	0	1	0	1	0	1	0	1						
RCW data	ACS=0	BYTE_CNT = 000000 (64 bytes)						CONT=1						
	SYS_ADDR[23-16] <sup>1</sup>													
	SYS_ADDR[15-8] <sup>1</sup>													
	SYS_ADDR[7-0] <sup>1</sup>													
	BYTE0													
	BYTE1													
	BYTE2													
	.....													
	BYTE63													
First pre-boot initialization command (optional)	ACS	BYTE_CNT						CONT=1						
	SYS_ADDR[23-16]													
	SYS_ADDR[15-8]													
	SYS_ADDR[7-0]													
	BYTE0													
	BYTE1													
	BYTE2													
	.....													
	BYTE N-1 (up to 63)													

Table continues on the next page...

## Functional description

**Table 29-3. Required format of data structure used by PBL (continued)**

	0	1	2	3	4	5	6	7
Second pre-boot initialization command (optional)	ACS			BYTE_CNT				CONT=1
				SYS_ADDR[23-16]				
				SYS_ADDR[15-8]				
				SYS_ADDR[7-0]				
				BYTE0				
				BYTE1				
				BYTE2				
				.....				
				BYTE N-1 (up to 63)				
				.....				
Last pre-boot initialization command (optional)	ACS			BYTE_CNT				CONT=1
				SYS_ADDR[23-16]				
				SYS_ADDR[15-8]				
				SYS_ADDR[7-0]				
				BYTE0				
				BYTE1				
				BYTE2				
				.....				
				BYTE N-1 (up to 63)				
				.....				
End command (required, special CRC check command with CONT=0)	ACS=0			BYTE_CNT = 000100 (4 bytes)				CONT=0
				SYS_ADDR[23:16] = 0x61 <sup>2</sup>				
				SYS_ADDR[15:8] = 0x00 <sup>2</sup>				
				SYS_ADDR[7:0] = 0x40 <sup>2</sup>				
				CRC0				
				CRC1				
				CRC2				
				CRC3				

1. SYS\_ADDR for the RCW should point to the DCFG\_CCSR\_RCWSRn register.
2. SYS\_ADDR = 0x61\_0040 is a PBI CRC check command (PBL block base address 0x161\_0000, but only 24-bits address are used here). See [Pre-boot initialization command loading by PBL](#) for more information.

The following table provides field descriptions for the PBL data structure.

**Table 29-4. PBL data structure field descriptions**

Field name	Description
ACS	Alternate Configuration Space. Logic 1 for alternate configuration space. Logic 0 for CCSR Space. Note that this field must be logic 0 for the RCW data.

*Table continues on the next page...*

**Table 29-4. PBL data structure field descriptions (continued)**

Field name	Description
BYTE_CNT	Byte Count. Represents the number of bytes associated with this address/data pair. Note that an encoding of all zeros represents 64 bytes. Only power of two multiples are legal (1, 2, 4, 8, 16, 32, and 64 bytes). The associated SYS_ADDR must be aligned to the byte count of the command.  000000 64 bytes 000001 1 byte 000010 2 bytes 000100 4 bytes 001000 8 bytes 010000 16 bytes 100000 32 bytes  All other encoding are reserved
CONT	Continue. This bit should be logic 1 for all commands except for the End command.
SYS_ADDR	System Address. The lowest order system address bits. Note that this permits addressability down to a byte for single byte transactions. SYS_ADDR must be aligned to the byte count of the command. The upper bits are either selected from alternate configuration BAR (depending on value of ACS) and then concatenated with SYS_ADDR to form the full address associated with the command.
BYTEn	Byte number $n$ where $n$ may range from 0 to 63. Byte 0 corresponds to address 0 relative to the SYS_ADDR (starting address), byte 1 for address 1,..., and byte 63 for address 63. Note the first command must be the RCW address/data pair and must be 64 bytes of data . Note that BYTEn is only present/valid in the data structure if $n$ is less than the BYTE_CNT.
CRCn	Cyclic Redundancy Check data. CRC value is calculated using CRC-32 algorithm with a start value of 0xFFFF_FFFF and an XOR with 0x0000_0000. The CRC covers all bytes stored in the ROM prior to the CRC. The polynomial used is  $1 + x^1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{12} + x^{16} + x^{22} + x^{23} + x^{26} + x^{32}$

## 29.4.4 RCW loading by PBL

If initialization of the source memory device completes without error, the PBL fetches RCW data from the source memory device and writes it to the RCW status registers (DCFG\_CCSR\_RCWSR $n$ ). See DCFG\_RCWSR for more details.

The PBL reports an error and sends a corresponding error code to the platform when one of the following conditions occurs:

- No preamble is detected.
- Alternate configuration space (ACS) is selected (ACS = 1).
- The byte count (BYTE\_CNT) is not 64 bytes.
- The system address (SYS\_ADDR) is not the specific address offset to the RCWSR to which the RCW contents are written.
- The system does not respond before internal time-out counter (32 bit) expires.

## 29.4.5 Pre-boot initialization command loading by PBL

If PBI is selected by the RCW[PBI\_SRC] field, the PBL PBI commands are processed and routed to CCSR, DDR, and other memory space.

The PBL reports an error and sends a corresponding error code to the platform when one of the following conditions occurs:

- CRC check error
- The byte count is not one of the supported values (1, 2, 4, 8, 16, 32, or 64 bytes). See [Table 29-4](#) for the supported encodings. Note that if byte count is 1 or 2 bytes, the PBL expects 3 or 2 bytes of padding (zeros) after the real data.
- The system address is misaligned to the byte count.
- The system address is in an offset range of protected CCSR space. See [CCSR registers blocked from PBL during secure boot](#), for more information.
- The system does not respond before internal time-out counter expires.
- Invalid internal PBL commands.
- Invalid End command, for example, not valid CRC command with CONT=0.

## 29.4.6 Reserved address space used as internal PBL commands

When a PBL command accesses the 64 KB (0x161\_0000-0x161\_FFFF) CCSR address space that is assigned to the PBL, the PBL treats this command entry as a PBL internal command.

Only 24 bits of address are used in PBL command, so the address for PBL command is 0x61\_0000 + offset. There are 4 bytes allocated for each command for command parameters. The byte count must be 4 bytes and unused data must be filled with 0s.

The following table defines each PBL command.

**Table 29-5. Description of PBL commands**

Command name	Offset	Parameter(s)	Command description
Flush	000	N/A	<p>Follows the previous write with a read from the same address. The purpose of this command is to ensure the previous write has taken effect.</p> <p><b>NOTE:</b> Use of the FLUSH command is restricted to CCSR space. Software should use the WAIT command after commands to non-CCSR space to allow them time to complete before issuing</p>

*Table continues on the next page...*

**Table 29-5. Description of PBL commands (continued)**

Command name	Offset	Parameter(s)	Command description
			subsequent commands to non-CCSR space.
CRC check	040	The 4 bytes of data indicate the cyclic redundancy check (CRC) value. CRC value is calculated using CRC-32 algorithm with a start value of 0xFFFF_FFFF and an XOR with 0x0000_0000.	Checks CRC for data blocks from the point where the last CRC check was performed until the data block before this CRC check command.
Jump	080	<p>The 4 bytes of data indicate the target address for the jump destination.</p> <p>For QuadSPI/eSDHC/NOR flash, this address is 4 bytes address relative to current address. The address after the jump is a 4 bytes aligned address.</p> <p>For 8 KB large page (128 pages per block) NAND flash, the higher 14 bits of address is non-bad block index relative to current block, the lower 18 bits are ignored. The address after jump is the start of target non-bad block.</p> <p>For 8 KB large page (64 pages per block) NAND flash, the higher 15 bits of address is non-bad block index relative to current block, the lower 17 bits are ignored. The address after jump is the start of target non-bad block.</p> <p>For 4 KB large page (128 pages per block) NAND flash, the higher 15 bits of address is non-bad block index relative to current block, the lower 17 bits are ignored. The address after jump is the start of target non-bad block.</p> <p>For 4 KB large page (64 pages per block) NAND flash, the higher 16 bits of address is non-bad block index relative to current block, the lower 16 bits are ignored. The address after jump is the start of target non-bad block.</p> <p>For 2 KB large page NAND flash, the higher 17 bits are used as a non-bad block index relative to current block, the lower 15 bits are ignored. The address after the jump is the start of target non-bad block.</p> <p>For small page NAND flash, the higher 20 bits are used as a non-bad block index relative to the current block, The lower 12 bits are ignored. The address after the jump is the start of target non-bad block.</p>	Upon receiving this command, the PBL reads next data from address starting from jump pointer indicated by the command parameter.
Wait	0C0	The 4 bytes of data indicate the number of cycles to wait. The clock source for the wait cycle is a PBL clock cycle (platform/2).	Upon receiving this command, the PBL waits a number of cycles indicated by the command parameter before reading the next data.

## 29.4.7 Error codes

The PBL reports errors to the platform under various conditions. PBL error codes are also sent along with the error indication.

The platform stores the error code in the DCFG\_CCSR\_RSTRQPBLSR[ERR\_CODE] bit. See DCFG\_RSTRQPBLSR for more information.

The following table shows the encoding for each pre-boot error condition.

**Table 29-6. Pre-boot error encoding**

Error code[0:6]	Error condition
0x00-0x0F	Reserved
0x10	Reserved
0x11-0x22	Reserved
0x23	IFC detected error in NAND_EVTER_STAT register, NAND_EVTER_STAT can be scanned out upon receiving this error.
0x24	IFC timed out
0x25-0x3F	Reserved
0x40	eSDHC: Err_Reset_1 Indicates that eSDHC did not complete its soft-reset sequence. SYSCTL[RSTA] (offset: 0x02C, mask: 0x0100_0000) did not clear within 1 second. The bit should clear when eSDHC completes its reset sequence.
0x40	eSDHC: Err_Reset_2 Indicates that eSDHC did not complete its soft-reset sequence. SYSCTL[RSTA] (offset: 0x02C, mask: 0x0100_0000) did not clear within 1 second. The bit should clear when eSDHC completes its reset sequence.
0x41	eSDHC: Err_Card_Ins Indicates that eSDHC has not detected an inserted card. PRSSTAT[CINS] (offset: 0x024, mask: 0x0001_0000) did not set within 1 second. The bit should set when eSDHC detects that a card has been inserted.
0x42	eSDHC: Err_Cmd_Data_Rdy Indicates that eSDHC did not detect an idle command and data interface. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD0_Cmd_Data_Rdy Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD0. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_SD_CMD8_Cmd_Data_Rdy Indicates that eSDHC did not detect an idle command and data interface before attempting to send SD CMD8. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
0x42	eSDHC: Err_CMD55_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD55. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_SD_ACMD41_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send SD ACMD41. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_MMC_CMD1_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send MMC CMD1. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD2_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD2. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_SD_CMD3_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send SD CMD3. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_MMC_CMD3_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send MMC CMD3. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD9_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD9. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD7_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD7. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD18_Cmd_Data_Rdy  Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD18. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x42	eSDHC: Err_CMD12_Cmd_Data_Rdy

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	Indicates that eSDHC did not detect an idle command and data interface before attempting to send CMD12. PRSSTAT[CDIHB] (offset: 0x024, mask: 0x0000_0002) and PRSSTAT[CIHB] (offset: 0x024, mask: 0x0000_0001) did not clear within 1 second. These bits should clear when eSDHC detects an idle command and data interface.
0x43	eSDHC: Err_Init_Clk
	Indicates that eSDHC did not complete its initialization clock sequence to the card. SYSCTL[INITA] (offset: 0x02C, mask: 0x0800_0000) did not clear within 1 second. The bit should clear when eSDHC completes transmission of initialization clocks to the card.
0x44	eSDHC: Err_CMD0_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending CMD0. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_SD_CMD8_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending SD CMD8. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD55_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending CMD55. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_SD_CMD41_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending SD ACMD41. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_MMC_CMD1_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending MMC CMD1. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD2_Cmd_Stat
	Indicates that eSDHC did not detect an error or command complete after sending CMD2. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_SD_CMD3_Cmd_Stat

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	Indicates that eSDHC did not detect an error or command complete after sending SD CMD3. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_MMC_CMD3_Cmd_Stat Indicates that eSDHC did not detect an error or command complete after sending MMC CMD3. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD9_Cmd_Stat Indicates that eSDHC did not detect an error or command complete after sending CMD9. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD7_Cmd_Stat Indicates that eSDHC did not detect an error or command complete after sending CMD7. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD18_Cmd_Stat Indicates that eSDHC did not detect an error or command complete after sending CMD18. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x44	eSDHC: Err_CMD12_Cmd_Stat Indicates that eSDHC did not detect an error or command complete after sending CMD12. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000), IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000), IRQSTAT[CC] (offset: 0x030, mask: 0x0000_0001) did not set within 1 second. These bits should set when eSDHC detects an error or command complete.
0x45	eSDHC: Err_CMD0_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending CMD0. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD8_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending SD CMD8. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD55_Cmd_Line_Conflict

Table continues on the next page...

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	Indicates that eSDHC detected a command line conflict after sending CMD55. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_SD_ACMD41_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending SD ACMD41. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_MMC_CMD1_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending MMC CMD1. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD2_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending CMD2. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_SD_CMD3_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending SD CMD3. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_MMC_CMD3_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending MMC CMD3. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD9_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending CMD9. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD7_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending CMD7. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x45	eSDHC: Err_CMD18_Cmd_Line_Conflict Indicates that eSDHC detected a command line conflict after sending CMD18. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
0x45	eSDHC: Err_CMD12_Cmd_Line_Conflict  Indicates that eSDHC detected a command line conflict after sending CMD12. IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) and IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bits should set when eSDHC detects a command line conflict. Both bits are set for the following reason. <ul style="list-style-type: none"><li>• Command Line Conflict Error (CCE and CTOE)</li></ul>
0x46	eSDHC: Err_SD_CMD8_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for SD CMD8. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons. <ul style="list-style-type: none"><li>• Command Index Error (CIE)</li><li>• Command End Bit Error (CEBE)</li><li>• Command CRC Error (CCE)</li></ul>
0x46	eSDHC: Err_CMD55_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for CMD55. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons. <ul style="list-style-type: none"><li>• Command Index Error (CIE)</li><li>• Command End Bit Error (CEBE)</li><li>• Command CRC Error (CCE)</li></ul>
0x46	eSDHC: Err_SD_ACMD41_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for SD ACMD41. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command End Bit Error (CEBE)</li></ul>
0x46	eSDHC: Err_MMC_CMD1_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for MMC CMD1. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command End Bit Error (CEBE)</li></ul>
0x46	eSDHC: Err_CMD2_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for CMD2. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000) or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons. <ul style="list-style-type: none"><li>• Command End Bit Error (CEBE)</li><li>• Command CRC Error (CCE)</li></ul>
0x46	eSDHC: Err_SD_CMD3_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for SD CMD3. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons. <ul style="list-style-type: none"><li>• Command Index Error (CIE)</li><li>• Command End Bit Error (CEBE)</li><li>• Command CRC Error (CCE)</li></ul>
0x46	eSDHC: Err_MMC_CMD3_Cmd_Rsp  Indicates that eSDHC detected an error with the command response for MMC CMD3. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons. <ul style="list-style-type: none"><li>• Command Index Error (CIE)</li></ul>

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	<ul style="list-style-type: none"> <li>• Command End Bit Error (CEBE)</li> <li>• Command CRC Error (CCE)</li> </ul>
0x46	<p>eSDHC: Err_CMD9_Cmd_Rsp</p> <p>Indicates that eSDHC detected an error with the command response for CMD9. IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000) or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons.</p> <ul style="list-style-type: none"> <li>• Command End Bit Error (CEBE)</li> <li>• Command CRC Error (CCE)</li> </ul>
0x46	<p>eSDHC: Err_CMD7_Cmd_Rsp</p> <p>Indicates that eSDHC detected an error with the command response for CMD7. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons.</p> <ul style="list-style-type: none"> <li>• Command Index Error (CIE)</li> <li>• Command End Bit Error (CEBE)</li> <li>• Command CRC Error (CCE)</li> </ul>
0x46	<p>eSDHC: Err_CMD18_Cmd_Rsp</p> <p>Indicates that eSDHC detected an error with the command response for CMD18. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons.</p> <ul style="list-style-type: none"> <li>• Command Index Error (CIE)</li> <li>• Command End Bit Error (CEBE)</li> <li>• Command CRC Error (CCE)</li> </ul>
0x46	<p>eSDHC: Err_CMD12_Cmd_Rsp</p> <p>Indicates that eSDHC detected an error with the command response for CMD12. IRQSTAT[CIE] (offset: 0x030, mask: 0x0008_0000), IRQSTAT[CEBE] (offset: 0x030, mask: 0x0004_0000), or IRQSTAT[CCE] (offset: 0x030, mask: 0x0002_0000) bit is set for one of the following reasons.</p> <ul style="list-style-type: none"> <li>• Command Index Error (CIE)</li> <li>• Command End Bit Error (CEBE)</li> <li>• Command CRC Error (CCE)</li> </ul>
0x47	<p>eSDHC: Err_SD_CMD8_Rsp_Data_Match</p> <p>Indicates that eSDHC did not properly communicate with the card due to a mismatch of the expected response of SD CMD8. CMDRSP0[VHS] (offset: 0x010, mask: 0x0000_0F00) was not equal to the configured SD coarse voltage or CMDRSP0[CHKPTRN] (offset: 0x010, mask: 0x0000_00FF) was not equal to the configured SD check pattern. Both fields are required to match in order to guarantee proper communication with the card.</p>
0x47	<p>eSDHC: Err_SD_ACMD41_Rsp_Data_Match</p> <p>Indicates that eSDHC did not properly communicate with the card due to a mismatch of the expected response of SD ACMD41. CMDRSP0[VDDVW] (offset: 0x010, mask: 0x00FF_FFFF) was not covered (matches at least one bit) by the configured SD fine voltage. At least one bit must match in order to guarantee proper communication with the card.</p>
0x47	<p>eSDHC: Err_MMC_CMD1_Rsp_Data_Match</p> <p>Indicates that eSDHC did not properly communicate with the card due to a mismatch of the expected response of MMC CMD1. CMDRSP0[VDDVW] (offset: 0x010, mask: 0x00FF_FFFF) was not covered (matches at least one bit) by the configured MMC voltage. At least one bit must match in order to guarantee proper communication with the card.</p>
0x48	eSDHC: Err_SD_ACMD41_Cmd_Rsp_Timeout

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	Indicates that eSDHC detected a timeout with the command response for SD ACMD41. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_MMC_CMD1_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for MMC CMD1. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_CMD2_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for CMD2. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_SD_CMD3_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for SD CMD3. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_MMC_CMD3_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for MMC CMD3. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_CMD9_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for CMD9. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_CMD7_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for CMD7. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_CMD18_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for CMD18. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x48	eSDHC: Err_CMD12_Cmd_Rsp_Timeout Indicates that eSDHC detected a timeout with the command response for CMD12. IRQSTAT[CTOE] (offset: 0x030, mask: 0x0001_0000) bit is set for the following reason. <ul style="list-style-type: none"><li>• Command Timeout Error (CTOE)</li></ul>
0x49	eSDHC: Err_SD_ACMD41_Rdy_Timeout Indicates that the card always indicates busy by examining the response of SD ACMD41. CMDRSP0[RDY] (offset: 0x010, mask: 0x8000_0000) did not set within 1 second. The bit should set when the card completes its initialization sequence.
0x49	eSDHC: Err_MMC_CMD1_Rdy_Timeout

*Table continues on the next page...*

**Table 29-6. Pre-boot error encoding (continued)**

Error code[0:6]	Error condition
	Indicates that the card always indicates busy by examining the response of MMC CMD1. CMDRSP0[RDY] (offset: 0x010, mask: 0x8000_0000) did not set within 1 second. The bit should set when the card completes its initialization sequence.
0x4A	eSDHC: Err_Rd_Buf_Rdy Indicates that eSDHC did not receive the watermark's amount (configured to 4 bytes) of data in its read buffer from CMD18. PRSSTAT[BREN] (offset: 0x024, mask: 0x0000_0800) did not set within 1 second. The bit should set when eSDHC receives the watermark's amount of data in its read buffer.
0x4B	eSDHC: Err_Data Indicates that eSDHC detected an error with the read data from CMD18. IRQSTAT[DEBE] (offset: 0x030, mask: 0x0040_0000) or IRQSTAT[DCE] (offset: 0x030, mask: 0x0020_0000) is set for one of the following reasons. <ul style="list-style-type: none"> <li>• Data End Bit Error (DEBE)</li> <li>• Data CRC Error (DCE)</li> </ul>
0x4C	eSDHC: Err_Data_Timeout Indicates that eSDHC detected a timeout with the read data from CMD18. IRQSTAT[DTOE] (offset: 0x030, mask: 0x0010_0000) is set for the following reason. <ul style="list-style-type: none"> <li>• Data Timeout Error (DTOE)</li> </ul>
0x4D	eSDHC: Err_Data_Reset Indicates that eSDHC did not complete its soft-reset sequence. SYSCTL[RSTD] (offset: 0x02C, mask: 0x0400_0000) did not clear within 1 second. The bit should clear when eSDHC completes its data reset sequence.
0x4E	eSDHC: Err_PBI_Timeout Indicates that pre-boot initialization (PBI) has not started within 1 second. An indication should be received when PBI starts.
0x4F	eSDHC: Reserved
0x50-0x6F	Reserved
0x70	No preamble is detected
0x71	ACS is logic 1 during RCW
0x72	Byte count is not 64 for RCW or 4 for PBL commands or 1/2/4/8/16/32/64 for pre-boot initialization commands
0x73	Misaligned address
0x74	Address is in protected space
0x75	CRC error
0x76	Time out counter expires
0x77	Unrecognized PBL commands, including unrecognized offset and invalid parameter.
0x78	Data transfer error from memory devices
0x79	Invalid end command error
0x7A	Invalid RCW or pre-boot initialization command source encoding
0x7B-0x7F	Reserved

## 29.5 Initialization/Application information

The PBL starts searching for the first good block. It begins to fetch the PBL image at the starting address of that block. It automatically searches for the next good block should the PBL image extend past one block.

If booting from NAND, the boot image must be placed on the next 4 KB boundary. The PBL searches for the next good block should that 4 KB boundary fall on a block boundary. The PBL then loads 4 KB into the buffer.

The PBL block search routine continues to search as necessary until reaching the end of the memory device.

### 29.5.1 Starting addresses

The following table lists the starting addresses of data fetched from each interface.

Note that in case of IFC NAND flash, the address starts from the first good block.

**Table 29-7. Starting addresses**

Interface	Starting address
IFC	0x0000_0000
QuadSPI	0x0000_0000
eSDHC	0x0000_1000

### 29.5.2 Software restrictions

Following are the restrictions for programming the PBI commands:

- Software must issue a Flush command after updating the alternate configuration space register (ALTCBAR) using PBI commands. This ensures that the logic has seen the update prior to any dependent writes being issued by the PBL.
- Use of the Flush command is restricted to CCSR space. Software must use the Wait command after commands to non-CCSR space to allow them time to complete before issuing subsequent commands to non-CCSR space.
- In case of IFC NAND FCM configuration, pages per block is determined by reading IFC\_CSOR0 register. This IFC register holds the pages per block information from control command. Control command coming from platform must contain the valid values at POR pins.

### 29.5.3 Software recommendations

Following are the recommendations for programming the PBL data structure:

- A CRC check command should always be placed immediately following the RCW command to ensure that a corrupted RCW is not consumed.

# Chapter 30

## Quad Serial Peripheral Interface (QuadSPI)

### 30.1 The QuadSPI module as implemented on the chip

This section provides details about how the QuadSPI module is implemented on the chip.

#### 30.1.1 LS1043A QuadSPI signals

The following table lists the SoC signal names and their corresponding QuadSPI module signal names used in this chapter:

**Table 30-1. LS1043A QuadSPI signals**

LS1043A signal name	QuadSPI module signal
QSPI_A_SCK	SCKFA
QSPI_B_SCK	SCKFB
QSPI_A_CS0	PCSFA1
QSPI_A_CS1	PCSFA2
QSPI_B_CS0	PCSFB1
QSPI_B_CS1	PCSFB2
QSPI_A_DATA[3:0]	IOFA
QSPI_B_DATA[3:0]	IOFB

#### 30.1.2 QuadSPI register reset values

All reset values for the QuadSPI registers that are device specific are shown in the following tables. All other register reset values are shown in the module memory map.

The QuadSPI module as implemented on the chip

**Table 30-2. QuadSPI Module Configuration Register Reset Values**

Register	Reset Value
Module Configuration Register (QuadSPI_MCR)	000F_4000h

**Table 30-3. QuadSPI Buffer Configuration Register Reset Values**

Register	Reset Value
Buffer0 Configuration Register (QuadSPI_BUF0CR)	0000_0000h
Buffer1 Configuration Register (QuadSPI_BUF1CR)	0000_0000h
Buffer2 Configuration Register (QuadSPI_BUF2CR)	0000_0000h
Buffer3 Configuration Register (QuadSPI_BUF3CR)	8000_0000h

**Table 30-4. Serial Flash A1 Top Address Reset Values**

Register	Reset Value
QuadSPI_SFA1AD	47FF_FC00h

**Table 30-5. Serial Flash A2 Top Address Reset Values**

Register	Reset Value
QuadSPI_SFA2AD	4FFF_FC00h

**Table 30-6. Serial Flash B1 Top Address Reset Values**

Register	Reset Value
QuadSPI_SFB1AD	57FF_FC00h

**Table 30-7. Serial Flash B2 Top Address Reset Values**

Register	Reset Value
QuadSPI_SFB2AD	5FFF_FC00h

### 30.1.3 QuadSPI\_MCR[SCLKCFG] mapping

The following table shows the QuadSPI serial clock configuration in the chip.

**Table 30-8. QuadSPI\_MCR[SCLKCFG] field description**

Field	Description
31-24	Serial Clock Configuration. Used for dividing clocks.

**Table 30-8. QuadSPI\_MCR[SCLKCFG] field description**

Field	Description
SCLKCFG	This field is reserved for the chip. QuadSPI clock division is performed by the SCFG_QSPI_CFG register. See <a href="#">QSPI CONFIG Register (SCFG_QSPI_CFG)</a> for different QuadSPI clock division settings.

### 30.1.4 QuadSPI boot initialization sequence

QSPI\_A\_CS0 is used for boot initialization. QSPI is only accessed in 1-bit mode and at low speed for RCW/PBI fetch. See [Table 30-10](#) for the RCW or PBI frequency of QSPI during this phase.

PBI can reconfigure the QSPI to higher frequency as required for the QSPI boot to occur faster.

### 30.1.5 LS1043A QuadSPI module special consideration

The QuadSPI module implements the following parameter settings in the chip:

**Table 30-9. LS1043A QuadSPI parameter settings**

QuadSPI parameters	LS1043A parameter value
Flexible AHB buffer size	1 KB
Multimaster mode support	Not supported, any references to that should be ignored.  Note: Multimaster mode is not supported so QuadSPI_BUFOCR[MSTRID] field is tied to zero and QuadSPI_BUFOCR[HP_EN] is not supported.
Stop mode support	Yes. Refers to LPM20 low power mode of the chip.
Internal Sampling	Not supported, any references to that should be ignored.
DQS sampling	Not supported, any references to that should be ignored.

QuadSPI is used as a pre-boot initialization device. The QuadSPI module supports two serial flash ports. Each port implements two chip-select enabling two separate serial flashes to be attached to individual port. See [QSPI CONFIG Register \(SCFG\\_QSPI\\_CFG\)](#) for QuadSPI configuration details.

**Table 30-10. QuadSPI clock divider options**

		RCW Phase		PBI Phase/Boot Phase	
SCFG_QSPI_CFG[CLK_SEL]	Divider value	SYSCLK (MHz)	Interface clock (MHz)	Core clock (MHz)	Interface clock (MHz)
0 (Default)	256	133	0.520	1200	4.7
1	64				18.8
2	32				37.5
3	24				50.0
4	20				60.0
5	16				75.0
6	12				100.0
7	8				150.0
9-F	Reserved				

## 30.2 Introduction

The Quad Serial Peripheral Interface (QuadSPI) block acts as an interface to one single or two external serial flash devices, each with up to four bidirectional data lines.

### 30.2.1 Features

The QuadSPI supports the following features:

- Flexible sequence engine to support various flash vendor devices.
- Single, dual, quad modes of operation.
- Two identical serial flash devices can be connected and accessed in parallel for data read operations, forming one (virtual) flash memory with doubled readout bandwidth.
- DMA support to read RX Buffer data via AMBA AHB bus (64-bit width interface) or IP registers space (32-bit access).
  - Inner loop size of DMA access can be configured.
- Multimaster accesses with priority
  - Flexible and configurable buffer for each master
- Multiple interrupt conditions (see [Table 30-20](#))

- Memory mapped read access to connected flash devices.
- Programmable sequence engine to cater to future command/protocol changes and able to support all existing vendor commands and operations.
  - Supports 3-byte and 4-byte addressing.

### 30.2.2 Block Diagram

The following figure is a block diagram of the Quad Serial Peripheral Interface (QuadSPI) module.

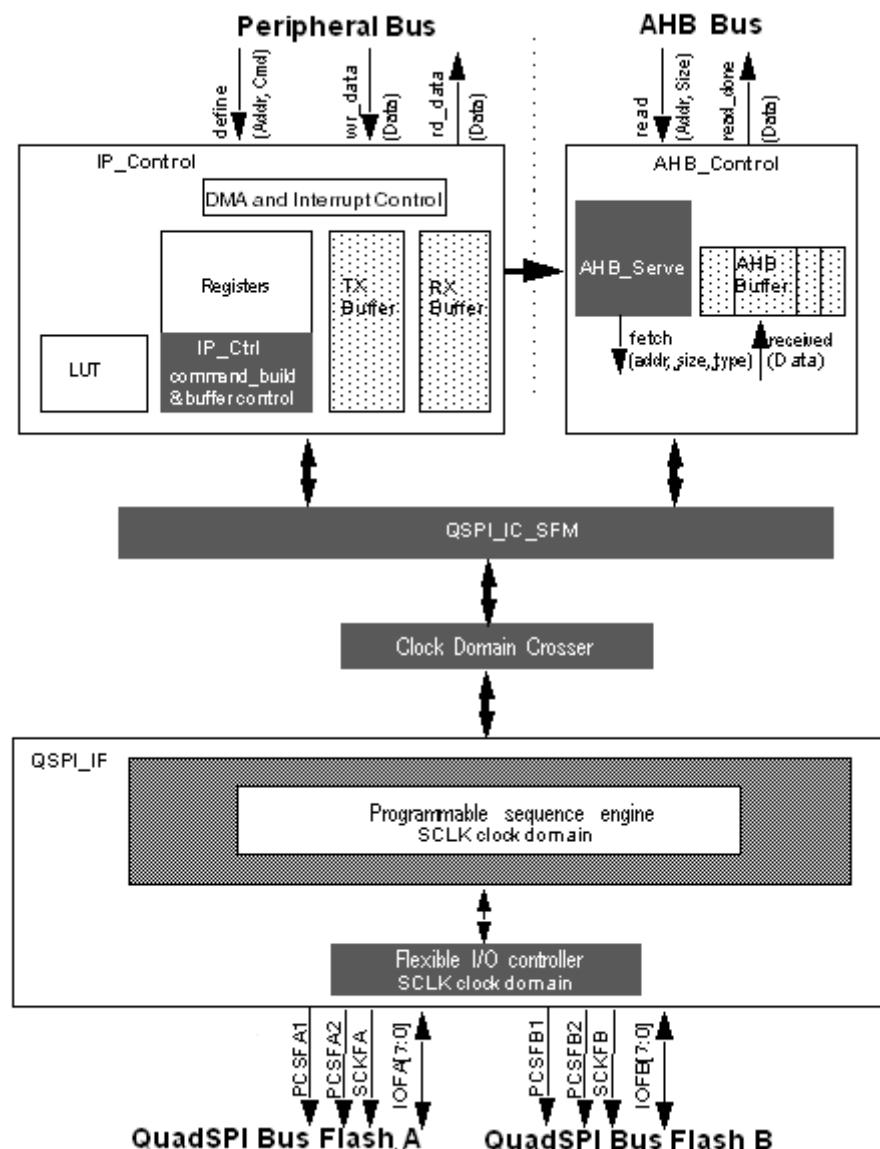


Figure 30-1. QuadSPI Block Diagram

### 30.2.3 QuadSPI Modes of Operation

This section provides information about the modes in which the QuadSPI module can be used.

#### 30.2.3.1 Normal Mode

In this mode, one or two external serial flash memory devices can be accessed. Further details about this mode of operation can be found in [Modes of Operation \(Normal Mode\)](#).

#### 30.2.3.2 Module Disable Mode

This mode is used for power management of the device containing the QuadSPI module. It is controlled by signals external to the QuadSPI. The clock to the non-memory mapped logic in the QuadSPI can be stopped while in Module Disable Mode.

#### 30.2.3.3 Stop Mode

This mode is also used for power management. When a request is made to enter Stop Mode, the QuadSPI block completes the action currently being processed, before the request is acknowledged.

### 30.2.4 Acronyms and Abbreviations

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

**Table 30-11. Acronyms and Abbreviations**

Terms	Description
AHB	Advanced High-performance Bus, version of AMBA
AMBA	Advanced Microcontroller Bus Architecture
BE	Big Endian Byte Ordering
CS	Chip Select
DMA	Direct Memory Access
MB	Megabyte. Each MB is 1024 * 1024 bytes
IFM	Individual Flash Mode
PFM	Parallel Flash Mode

*Table continues on the next page...*

**Table 30-11. Acronyms and Abbreviations (continued)**

Terms	Description
LSB	Least Significant Bit
MSB	Most Significant Bit
PCS	Peripheral Chip Select
QSPI, QuadSPI	Quad Serial Peripheral Interface
SCK	Serial Communications Clock
w1c	Write 1 to clear, writing a 1 to this field resets the flag

### 30.2.5 Glossary for QuadSPI module

**Table 30-12. Glossary**

Term	Definition
AHB Command	An AHB Command is a SFM Command triggered by a read access to the address range belonging to the memory mapped access defined in <a href="#">Table 30-17</a> . Refer to <a href="#">AHB Commands</a> for details.
Asserted	A signal that is asserted is in its active state. An active low signal changes from logic level one to logic level zero when asserted, and an active high signal changes from logic level zero to logic level one.
Clear	To clear a bit or bits means to establish logic level zero on the bit or bits.
Clock Phase	Determines when the data should be sampled relative to the active edge of SCK
Clock Polarity	Determines the idle state of the SCK signal.
Drain	To remove entries from a FIFO by software or hardware.
Endianness	Byte Ordering scheme.
Field	Two or more register bits grouped together.
Fill	To add entries to a FIFO by software or hardware.
Host	Refers to another functional block in the device containing the QuadSPI module
Instruction Code	8 bits defining the type of command to be executed.
IP Command	An IP Command is a SFM Command triggered by writing into the QSPI_IPCR[SEQID] field.
Logic level one	The voltage that corresponds to Boolean true (1) state.
Logic level zero	The voltage that corresponds to Boolean false (0) state.
Negated	A signal that is negated is in its inactive state. An active low signal changes from logic level 0 to logic level 1 when negated, and an active high signal changes from logic level 1 to logic level 0.
QSPI_AMBA_BASE	First address of QuadSPI address space on system memory map.
QSPI_ARDB_BASE	First address of QuadSPI Rx Buffer on system memory map.
Set	To set a bit or bits means to establish logic level one on the bit or bits.
RX Buffer PUSH Event	Addition of valid entries into the RX Buffer. In the default case each Buffer PUSH Event adds 2 entries to the RX Buffer since the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash device it is possible for the very last Buffer PUSH Event that only one entry is added.  The QSPI_RBSR[RDBFL] field is incremented by the number of entries added to the RX Buffer.

*Table continues on the next page...*

**Table 30-12. Glossary (continued)**

Term	Definition
RX Buffer POP Event	Removal of valid entries from the RX Buffer. Each Buffer POP Event removes (QSPI_RBCT[WMRK] + 1) valid entries from the buffer. The QSPI_RBSR[RDBFL] field is decremented by the same number and the QSPI_RBSR[RDCTR] field is incremented accordingly.
Individual Flash Mode	Access to a single, individual serial flash device. Refer to <a href="#">Serial Flash Access Schemes</a> for details.
Parallel Flash Mode	Read access to two serial flash devices attached to the QuadSPI module in parallel. Refer to <a href="#">Serial Flash Access Schemes</a> for details.
SFM Command	Serial Flash Memory Command. A SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash or results in an error. Refer to <a href="#">Table 30-30</a> for details on errors.
Single/Dual/Quad Instructions	Depending on the serial flash device connected to the QuadSPI module there will be instructions using a different number of data lines. <ul style="list-style-type: none"> <li>• Single: Single line I/O with one data out and one data in line to/from the serial flash device.</li> <li>• Dual: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash device or the QuadSPI module</li> <li>• Quad: Quad line I/O with 4 bidirectional I/O lines, driven alternatively by the serial flash device or the QuadSPI</li> </ul>
Transaction	A transaction consists of all flags, data and signals in either direction to execute a command for an attached serial flash device. It is a combination of chip select, sclk, instruction code, address, mode-and/or dummy bytes, transmit and/or receive data.
LUT	Look-up table.

### 30.3 External Signal Description

This section provides the external signal information of the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

**Table 30-13. Signal Properties**

Signal Name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash A1	O	This signal is the chip select for the serial flash device A1. A1 represents the first device in a dual-die package flash A or the first of the two flash devices that share IOFA. Refer to <a href="#">Dual Die Flashes</a> for more details.
PCSFA2	Peripheral Chip Select Flash A2	O	This signal is the chip select for the serial flash device A2. A2 represents the second device in a dual-die package flash A or the second of the two flash devices that share IOFA. Refer to <a href="#">Dual Die Flashes</a> for more details.
PCSFB1	Peripheral Chip Select Flash B1	O	This signal is the chip select for the serial flash device B1. B1 represents the first device in a dual-die package flash B or the first of the two flash devices that share IOFB. Refer to <a href="#">Dual Die Flashes</a> for more details.
PCSFB2	Peripheral Chip Select Flash B2	O	This signal is the chip select for the serial flash device B2. B2 represents the second device in a dual-die package flash B or the second of the two flash devices that share IOFB. Refer to <a href="#">Dual Die Flashes</a> for more details.

*Table continues on the next page...*

**Table 30-13. Signal Properties (continued)**

Signal Name	Function	Direction	Description
SCKFA	Serial Clock Flash A	O	This signal is the serial clock output to the serial flash device A.
SCKFB	Serial Clock Flash B	O	This signal is the serial clock output to the serial flash device B.
IOFA[3:0]	Serial I/O Flash A	I/O	These signals are the data I/O lines to/from the serial flash device A. Refer to <a href="#">Driving External Signals</a> for details about the signal drive and timing behavior. Note that the signal pins of the serial flash device may change their function according to the SFM Command executed, leaving them as control inputs when Single and Dual Instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].
IOFB[3:0]	Serial I/O Flash B	I/O	These signals are the data I/O lines to/from the serial flash device B. Refer to <a href="#">Driving External Signals</a> for details about the signal drive and timing behavior. Note that the signal pins of the serial flash device may change their function according to the SFM Command executed, leaving them as control inputs when Single and Dual Instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFB[0] and expects data on IOFB[1].

### 30.3.1 Driving External Signals

The different phases of the serial flash access scheme are shown in the following figure.

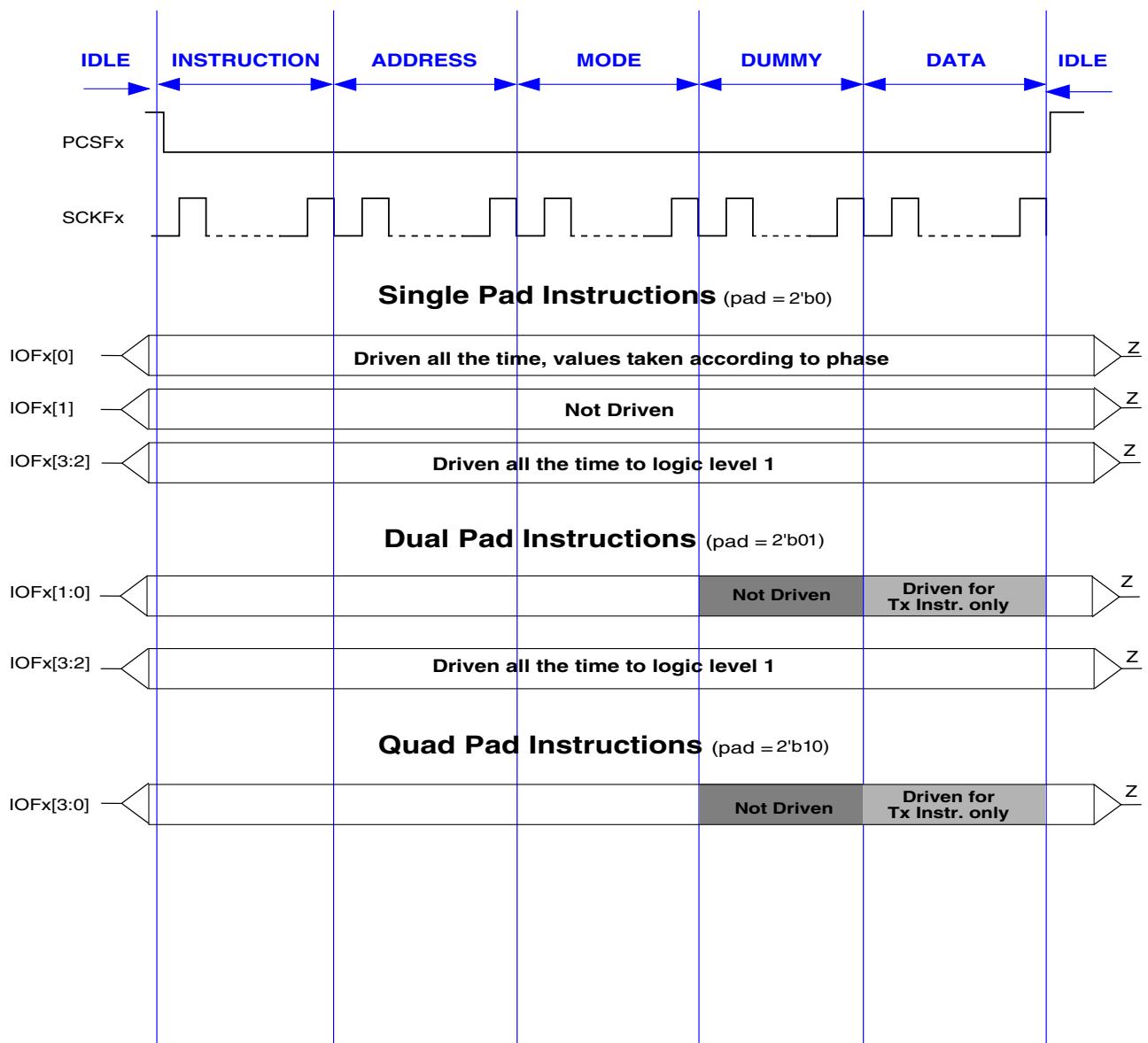


Figure 30-2. Serial Flash Access Scheme

The different phases and the I/O driving characteristics of the QuadSPI module are characterized in the following way:

- **IDLE:** Serial flash device not selected. No interaction with the serial flash device. All IOFx signals driven to High-Z.
- **INSTRUCTION:** Serial flash device selected. The instruction is sent to the serial flash device. All IOFx signals are driven.
- **ADDRESS:** Serial Flash Address is sent to the device. All IOFx signals are driven. Note that this phase is not applicable for all SFM Commands.

- **MODE**: Mode bytes are sent to the serial flash device. All IOFx signals are driven. Note that this phase is not applicable for all SFM Commands.
- **DUMMY**: Dummy clocks are provided to the serial flash device. Refer to the [Figure 30-2](#) for the IOFx signals driven. The actual data lines required for the SFM Command executed are not driven for data read commands. Note that this phase is not applicable for all SFM Commands.
- **DATA**: Serial flash data are sent to or received from the serial flash device. Refer to the preceding figure for the IOFx signals driven. The actual data lines required for the SFM Command executed are not driven for data read commands. Note that this phase is not applicable for all SFM Commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases.

In Individual Flash Mode this applies to the selected flash device. In Parallel Flash Mode this applies to both serial flash devices simultaneously.

## 30.4 Memory Map and Register Definition

This section provides the memory map and register definitions of the QuadSPI module.

### 30.4.1 Register Write Access

This section describes the write access restriction terms that apply to all registers, which can be one of the following:

- **Register Write Access Restriction**

For each register bit and register field, the write access conditions are specified in the detailed register description. A description of the write access conditions is given in the following table. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

**Table 30-14. Register Write Access Restrictions**

Condition	Description
Anytime	No write access restriction.
Disabled Mode	Write access only if <i>QSPI_MCR[MDIS]</i> = 1.
Normal Mode	Write access only if the module is in <i>Normal Mode</i> .

- **Register Write Access Requirements**

All registers can be accessed with 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16/32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each register affected.

### 30.4.2 Peripheral Bus Register Descriptions

This section provides the memory map and register definitions of the QuadSPI module.

**QuadSPI memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
155_0000	Module Configuration Register (QuadSPI_MCR)	32	R/W	<a href="#">See section</a>	30.4.2.1/ 1668
155_0008	IP Configuration Register (QuadSPI_IPCR)	32	R/W	0000_0000h	30.4.2.2/ 1670
155_000C	Flash Configuration Register (QuadSPI_FLSHCR)	32	R/W	0000_0303h	30.4.2.3/ 1671
155_0010	Buffer0 Configuration Register (QuadSPI_BUF0CR)	32	R/W	<a href="#">See section</a>	30.4.2.4/ 1672
155_0014	Buffer1 Configuration Register (QuadSPI_BUF1CR)	32	R/W	<a href="#">See section</a>	30.4.2.5/ 1673
155_0018	Buffer2 Configuration Register (QuadSPI_BUF2CR)	32	R/W	<a href="#">See section</a>	30.4.2.6/ 1674
155_001C	Buffer3 Configuration Register (QuadSPI_BUF3CR)	32	R/W	<a href="#">See section</a>	30.4.2.7/ 1675
155_0020	Buffer Generic Configuration Register (QuadSPI_BFGENCR)	32	R/W	0000_0000h	30.4.2.8/ 1676
155_0030	Buffer0 Top Index Register (QuadSPI_BUF0IND)	32	R/W	0000_0000h	30.4.2.9/ 1677

Table continues on the next page...

**QuadSPI memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
155_0034	Buffer1 Top Index Register (QuadSPI_BUF1IND)	32	R/W	0000_0000h	<a href="#">30.4.2.10/ 1677</a>
155_0038	Buffer2 Top Index Register (QuadSPI_BUF2IND)	32	R/W	0000_0000h	<a href="#">30.4.2.11/ 1678</a>
155_0100	Serial Flash Address Register (QuadSPI_SFAR)	32	R/W	0000_0000h	<a href="#">30.4.2.12/ 1679</a>
155_0108	Sampling Register (QuadSPI_SMPR)	32	R/W	0000_0000h	<a href="#">30.4.2.13/ 1679</a>
155_010C	RX Buffer Status Register (QuadSPI_RBSR)	32	R	0000_0000h	<a href="#">30.4.2.14/ 1681</a>
155_0110	RX Buffer Control Register (QuadSPI_RBCT)	32	R/W	0000_0000h	<a href="#">30.4.2.15/ 1681</a>
155_0150	TX Buffer Status Register (QuadSPI_TBSR)	32	R	0000_0000h	<a href="#">30.4.2.16/ 1682</a>
155_0154	TX Buffer Data Register (QuadSPI_TBDR)	32	R/W	0000_0000h	<a href="#">30.4.2.17/ 1683</a>
155_015C	Status Register (QuadSPI_SR)	32	R	0000_3800h	<a href="#">30.4.2.18/ 1684</a>
155_0160	Flag Register (QuadSPI_FR)	32	w1c	0800_0000h	<a href="#">30.4.2.19/ 1686</a>
155_0164	Interrupt and DMA Request Select and Enable Register (QuadSPI_RSER)	32	R/W	0000_0000h	<a href="#">30.4.2.20/ 1689</a>
155_0168	Sequence Suspend Status Register (QuadSPI_SPNDST)	32	R	0000_0000h	<a href="#">30.4.2.21/ 1693</a>
155_016C	Sequence Pointer Clear Register (QuadSPI_SPTRCLR)	32	R/W	0000_0000h	<a href="#">30.4.2.22/ 1695</a>
155_0180	Serial Flash A1 Top Address (QuadSPI_SFA1AD)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.23/ 1696</a>
155_0184	Serial Flash A2 Top Address (QuadSPI_SFA2AD)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.24/ 1696</a>
155_0188	Serial Flash B1Top Address (QuadSPI_SFB1AD)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.25/ 1697</a>
155_018C	Serial Flash B2Top Address (QuadSPI_SFB2AD)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.26/ 1697</a>
155_0200	RX Buffer Data Register (QuadSPI_RBDR0)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0204	RX Buffer Data Register (QuadSPI_RBDR1)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0208	RX Buffer Data Register (QuadSPI_RBDR2)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_020C	RX Buffer Data Register (QuadSPI_RBDR3)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0210	RX Buffer Data Register (QuadSPI_RBDR4)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>

*Table continues on the next page...*

## QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
155_0214	RX Buffer Data Register (QuadSPI_RBDR5)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0218	RX Buffer Data Register (QuadSPI_RBDR6)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_021C	RX Buffer Data Register (QuadSPI_RBDR7)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0220	RX Buffer Data Register (QuadSPI_RBDR8)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0224	RX Buffer Data Register (QuadSPI_RBDR9)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0228	RX Buffer Data Register (QuadSPI_RBDR10)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_022C	RX Buffer Data Register (QuadSPI_RBDR11)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0230	RX Buffer Data Register (QuadSPI_RBDR12)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0234	RX Buffer Data Register (QuadSPI_RBDR13)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0238	RX Buffer Data Register (QuadSPI_RBDR14)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_023C	RX Buffer Data Register (QuadSPI_RBDR15)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0240	RX Buffer Data Register (QuadSPI_RBDR16)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0244	RX Buffer Data Register (QuadSPI_RBDR17)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0248	RX Buffer Data Register (QuadSPI_RBDR18)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_024C	RX Buffer Data Register (QuadSPI_RBDR19)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0250	RX Buffer Data Register (QuadSPI_RBDR20)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0254	RX Buffer Data Register (QuadSPI_RBDR21)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0258	RX Buffer Data Register (QuadSPI_RBDR22)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_025C	RX Buffer Data Register (QuadSPI_RBDR23)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0260	RX Buffer Data Register (QuadSPI_RBDR24)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0264	RX Buffer Data Register (QuadSPI_RBDR25)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0268	RX Buffer Data Register (QuadSPI_RBDR26)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>

Table continues on the next page...

**QuadSPI memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
155_026C	RX Buffer Data Register (QuadSPI_RBDR27)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0270	RX Buffer Data Register (QuadSPI_RBDR28)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0274	RX Buffer Data Register (QuadSPI_RBDR29)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0278	RX Buffer Data Register (QuadSPI_RBDR30)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_027C	RX Buffer Data Register (QuadSPI_RBDR31)	32	R	0000_0000h	<a href="#">30.4.2.27/ 1698</a>
155_0300	LUT Key Register (QuadSPI_LUTKEY)	32	R/W	5AF0_5AF0h	<a href="#">30.4.2.28/ 1699</a>
155_0304	LUT Lock Configuration Register (QuadSPI_LCKCR)	32	R/W	0000_0002h	<a href="#">30.4.2.29/ 1699</a>
155_0310	Look-up Table register (QuadSPI_LUT0)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0314	Look-up Table register (QuadSPI_LUT1)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0318	Look-up Table register (QuadSPI_LUT2)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_031C	Look-up Table register (QuadSPI_LUT3)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0320	Look-up Table register (QuadSPI_LUT4)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0324	Look-up Table register (QuadSPI_LUT5)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0328	Look-up Table register (QuadSPI_LUT6)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_032C	Look-up Table register (QuadSPI_LUT7)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0330	Look-up Table register (QuadSPI_LUT8)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0334	Look-up Table register (QuadSPI_LUT9)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0338	Look-up Table register (QuadSPI_LUT10)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_033C	Look-up Table register (QuadSPI_LUT11)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0340	Look-up Table register (QuadSPI_LUT12)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0344	Look-up Table register (QuadSPI_LUT13)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>
155_0348	Look-up Table register (QuadSPI_LUT14)	32	R/W	<a href="#">See section</a>	<a href="#">30.4.2.30/ 1700</a>

*Table continues on the next page...*

## QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
155_034C	Look-up Table register (QuadSPI_LUT15)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0350	Look-up Table register (QuadSPI_LUT16)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0354	Look-up Table register (QuadSPI_LUT17)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0358	Look-up Table register (QuadSPI_LUT18)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_035C	Look-up Table register (QuadSPI_LUT19)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0360	Look-up Table register (QuadSPI_LUT20)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0364	Look-up Table register (QuadSPI_LUT21)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0368	Look-up Table register (QuadSPI_LUT22)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_036C	Look-up Table register (QuadSPI_LUT23)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0370	Look-up Table register (QuadSPI_LUT24)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0374	Look-up Table register (QuadSPI_LUT25)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0378	Look-up Table register (QuadSPI_LUT26)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_037C	Look-up Table register (QuadSPI_LUT27)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0380	Look-up Table register (QuadSPI_LUT28)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0384	Look-up Table register (QuadSPI_LUT29)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0388	Look-up Table register (QuadSPI_LUT30)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_038C	Look-up Table register (QuadSPI_LUT31)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0390	Look-up Table register (QuadSPI_LUT32)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0394	Look-up Table register (QuadSPI_LUT33)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0398	Look-up Table register (QuadSPI_LUT34)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_039C	Look-up Table register (QuadSPI_LUT35)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03A0	Look-up Table register (QuadSPI_LUT36)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700

Table continues on the next page...

**QuadSPI memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
155_03A4	Look-up Table register (QuadSPI_LUT37)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03A8	Look-up Table register (QuadSPI_LUT38)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03AC	Look-up Table register (QuadSPI_LUT39)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03B0	Look-up Table register (QuadSPI_LUT40)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03B4	Look-up Table register (QuadSPI_LUT41)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03B8	Look-up Table register (QuadSPI_LUT42)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03BC	Look-up Table register (QuadSPI_LUT43)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03C0	Look-up Table register (QuadSPI_LUT44)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03C4	Look-up Table register (QuadSPI_LUT45)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03C8	Look-up Table register (QuadSPI_LUT46)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03CC	Look-up Table register (QuadSPI_LUT47)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03D0	Look-up Table register (QuadSPI_LUT48)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03D4	Look-up Table register (QuadSPI_LUT49)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03D8	Look-up Table register (QuadSPI_LUT50)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03DC	Look-up Table register (QuadSPI_LUT51)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03E0	Look-up Table register (QuadSPI_LUT52)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03E4	Look-up Table register (QuadSPI_LUT53)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03E8	Look-up Table register (QuadSPI_LUT54)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03EC	Look-up Table register (QuadSPI_LUT55)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03F0	Look-up Table register (QuadSPI_LUT56)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03F4	Look-up Table register (QuadSPI_LUT57)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_03F8	Look-up Table register (QuadSPI_LUT58)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700

*Table continues on the next page...*

## QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
155_03FC	Look-up Table register (QuadSPI_LUT59)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0400	Look-up Table register (QuadSPI_LUT60)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0404	Look-up Table register (QuadSPI_LUT61)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_0408	Look-up Table register (QuadSPI_LUT62)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700
155_040C	Look-up Table register (QuadSPI_LUT63)	32	R/W	<a href="#">See section</a>	30.4.2.30/ 1700

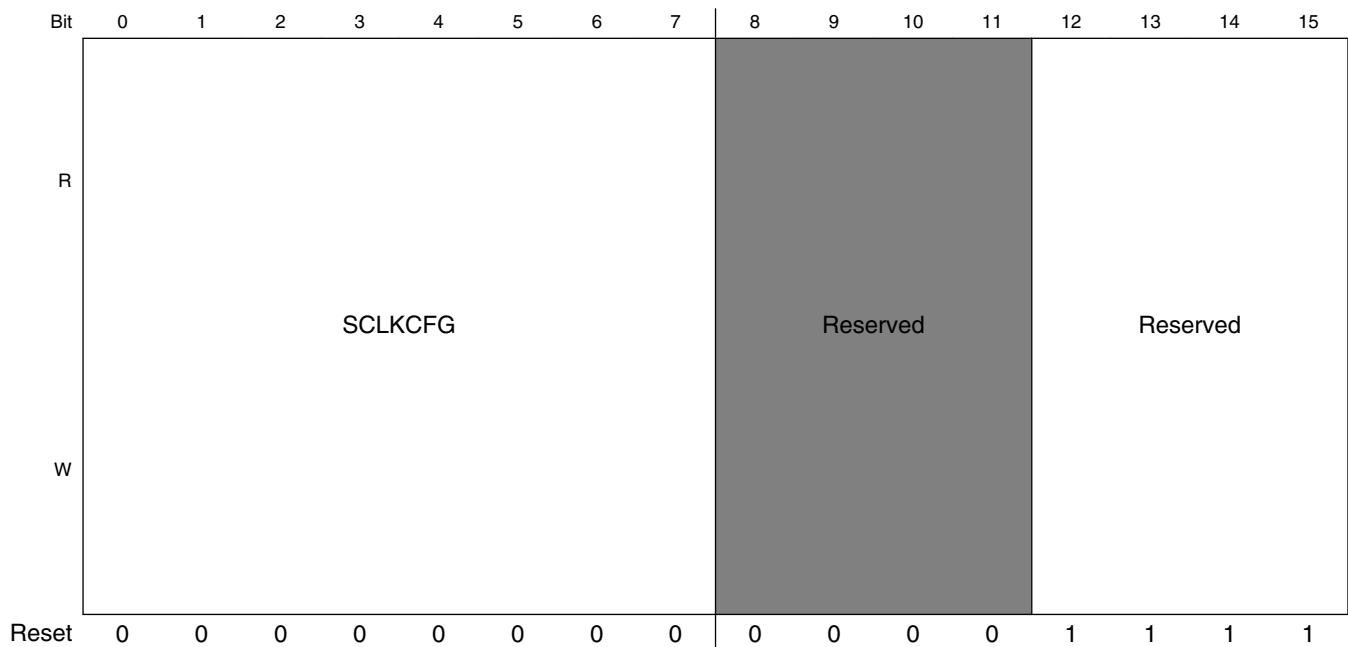
**30.4.2.1 Module Configuration Register (QuadSPI\_MCR)**

The QuadSPI\_MCR holds configuration data associated with QuadSPI operation.

*Write:*

- All other fields: Anytime

Address: 155\_0000h base + 0h offset = 155\_0000h



Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R					0	0										
DOZE		MDIS	Reserved					Reserved		Reserved		Reserved		END_CFG	SWRSTHD	SWRSTSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	0	0	0	*	*	0	0

\* Notes:

- END\_CFG field: See the module configuration for the device specific reset value.

### QuadSPI\_MCR field descriptions

Field	Description
0–7 SCLKCFG	Serial Clock Configuration. This field configuration is chip specific. For details, refer to chip-specific QuadSPI information. It may be used for dividing clocks.
8–11 Reserved	This field is reserved.
12–15 Reserved	This field is reserved. This field is reserved and should always be set to 0xF.
16 DOZE	Doze Enable. The DOZE bit provides support for externally controlled Doze Mode power-saving mechanism.  0 A doze request will be ignored by the QuadSPI module 1 A doze request will be processed by the QuadSPI module
17 MDIS	Module Disable. The MDIS bit allows the clock to the non-memory mapped logic in the QuadSPI to be stopped, putting the QuadSPI in a software controlled power-saving state.  0 Enable QuadSPI clocks. 1 Allow external logic to disable QuadSPI clocks.
18–19 Reserved	This field is reserved.
20 CLR_TXF	Clear TX FIFO/Buffer. Invalidates the TX Buffer content. This is a self-clearing field.  0 No action. 1 Read and write pointers of the TX Buffer are reset to 0. QSPI_TBSR[TRCTR] is reset to 0.

Table continues on the next page...

### QuadSPI\_MCR field descriptions (continued)

Field	Description
21 CLR_RXF	<p>Clear RX FIFO. Invalidates the RX Buffer.</p> <p>This is a self-clearing field.</p> <p>0 No action. 1 Read and write pointers of the RX Buffer are reset to 0. QSPI_RBSR[RDBFL] is reset to 0.</p>
22–23 Reserved	This field is reserved.
24 Reserved	This field is reserved.
25 Reserved	This field is reserved.
26–27 Reserved	This field is reserved.
28–29 END_CFG	Defines the endianness of the QuadSPI module. For more details refer to <a href="#">Byte Ordering Endianess</a>
30 SWRSTHD	<p>Software reset for AHB domain</p> <p>0 No action 1 AHB domain flops are reset. Does not reset configuration registers.</p> <p>It is advisable to reset both the serial flash domain and AHB domain at the same time. Resetting only one domain might lead to side effects.</p> <p><b>NOTE:</b> The software resets need the clock to be running to propagate to the design. The MCR[MDIS] should therefore be set to 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the MCR[MDIS] bit to 1. Once the software resets have been deasserted, the normal operation can be started by setting the MCR[MDIS] bit to 0.</p>
31 SWRSTSD	<p>Software reset for serial flash domain</p> <p>0 No action 1 Serial Flash domain flops are reset. Does not reset configuration registers.</p> <p>It is advisable to reset both the serial flash domain and AHB domain at the same time. Resetting only one domain might lead to side effects.</p> <p><b>NOTE:</b> The software resets need the clock to be running to propagate to the design. The MCR[MDIS] should therefore be set to 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTSD] to 0), it is recommended to set the MCR[MDIS] bit to 1. Once the software resets have been deasserted, the normal operation can be started by setting the MCR[MDIS] bit to 0.</p>

#### 30.4.2.2 IP Configuration Register (QuadSPI\_IPCR)

The IP configuration register provides all the configuration required for an IP initiated command. An IP command can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash is started as per the sequence pointed to by the SEQID field. Refer to [Normal Mode](#), for details about the command triggering and command execution.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$

Address: 155\_0000h base + 8h offset = 155\_0008h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved				SEQID				Reserved							PAR_EN
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	IDATSZ															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### QuadSPI\_IPCR field descriptions

Field	Description
0–3 Reserved	This field is reserved.
4–7 SEQID	Points to a sequence in the Look-up table. This field defines bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. Refer to <a href="#">Look-up Table</a> for more details. A write to this field triggers a transaction on the serial flash interface.
8–14 Reserved	This field is reserved.
15 PAR_EN	When set, a transaction to two serial flash devices is triggered in parallel mode. Refer to <a href="#">Parallel Flash Mode</a> for more details.
16–31 IDATSZ	IP data transfer size. Defines the data transfer size in bytes of the IP command.

### 30.4.2.3 Flash Configuration Register (QuadSPI\_FLSHCR)

The Flash configuration register contains the flash device specific timings that must be met by the QuadSPI controller for the device to function correctly.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$
- $QSPI\_SR[IP\_ACC] = 0$

Address: 155\_0000h base + Ch offset = 155\_000Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															TCSH	Reserved				TCSS											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	

**QuadSPI\_FLSHCR field descriptions**

Field	Description
0–19 -	This field is reserved. Reserved.
20–23 TCSH	Serial flash CS hold time in terms of serial flash clock cycles. <b>NOTE:</b> The actual delay between chip select and clock is defined as: <ul style="list-style-type: none"> <li>• TCSH = 1 SCK clk if N= 0/1 else, N SCK clk if N&gt;1, where N is the setting of TCSH</li> </ul>
24–27 Reserved	This field is reserved. Reserved.
28–31 TCSS	Serial flash CS setup time in terms of serial flash clock cycles. <b>NOTE:</b> <ul style="list-style-type: none"> <li>1. The actual delay between chip select and clock is defined as:<ul style="list-style-type: none"> <li>• TCSS = 0.5 SCK clk if N= 0/1 else, N+0.5 SCK clk if N&gt;1, where N is the setting of TCSS.</li> </ul></li> <li>2. Any update to the TCSS register bits is visible on the flash interface only from the second transaction following the update.</li> </ul>

**30.4.2.4 Buffer0 Configuration Register (QuadSPI\_BUFOCR)**

This register provides the configuration for any access to buffer0. An access is routed to buffer0 when the master port number of the incoming AHB request matches the MSTRID field of BUFOCR. Any buffer "miss" leads to a serial flash transaction being triggered as per the sequence pointed to the SEQID field. Buffer0 may also be configured as a high priority buffer by setting the HP\_EN field of this register.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 10h offset = 155\_0010h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	HP_	EN	Reserved													
W	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ADATSZ								Reserved				MSTRID			
W	Reset	0	0	0	0	0	0	0	0	0	0	0	*	*	*	*

\* Notes:

- MSTRID field: See the module configuration for reset value.

**QuadSPI\_BUFOCR field descriptions**

Field	Description
0 HP_EN	High Priority Enable. When set, the master/projects/Modules/D_IP_QUADSPI_Vybris/topics/flexible_ahb_buffers associated with this buffer is assigned a priority higher than the rest of the masters.

*Table continues on the next page...*

**QuadSPI\_BUFOCR field descriptions (continued)**

Field	Description
	An access by a high priority master will suspend any ongoing prefetch by another AHB master and will be serviced on high priority. Refer to <a href="#">Flexible AHB Buffers</a> for details.
1–15 Reserved	This field is reserved.
16–23 ADATSZ	AHB data transfer size  Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash. For example, a value of 0x2 will set transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
24–27 Reserved	This field is reserved.
28–31 MSTRID	Master ID. The ID of the AHB master associated with BUFFER0. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

**30.4.2.5 Buffer1 Configuration Register (QuadSPI\_BUFI1CR)**

This register provides the configuration for any access to buffer1. An access is routed to buffer1 when the master port number of the incoming AHB request matches the MSTRID field of BUFI1CR. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 14h offset = 155\_0014h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*	*	*

\* Notes:

- MSTRID field: See the module configuration for reset value.

**QuadSPI\_BUFI1CR field descriptions**

Field	Description
0–15 Reserved	This field is reserved.
16–23 ADATSZ	AHB data transfer size  Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash. For example, a value of 0x2 will set transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.

*Table continues on the next page...*

**QuadSPI\_BUF1CR field descriptions (continued)**

Field	Description
24–27 Reserved	This field is reserved.
28–31 MSTRID	Master ID. The ID of the AHB master associated with BUFFER1. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

**30.4.2.6 Buffer2 Configuration Register (QuadSPI\_BUF2CR)**

This register provides the configuration for any access to buffer2. An access is routed to buffer2 when the master port number of the incoming AHB request matches the MSTRID field of BUF2CR. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 18h offset = 155\_0018h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 \* \* \* \*

\* Notes:

- MSTRID field: See the module configuration for the device specific reset value.

**QuadSPI\_BUF2CR field descriptions**

Field	Description
0–15 Reserved	This field is reserved. Reserved.
16–23 ADATSZ	AHB data transfer size  Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash. For example, a value of 0x2 will set transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
24–27 Reserved	This field is reserved. Reserved.
28–31 MSTRID	Master ID. The ID of the AHB master associated with BUFFER2. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

### 30.4.2.7 Buffer3 Configuration Register (QuadSPI\_BUFB3CR)

This register provides the configuration for any access to buffer3. An access is routed to buffer3 when the master port number of the incoming AHB request matches the MSTRID field of BUFB3CR. Any buffer "miss" leads to the buffer being flushed a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

In the case that the ALLMST field is not set, any such transaction (where master port number does not match any of the MSTRID fields) will be considered as an illegal programming. This kind of programming is not allowed.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 1Ch offset = 155\_001Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	ALLMST	Reserved															
W																	
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	ADATSZ								Reserved				MSTRID				
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	*	*	*	*	

\* Notes:

- MSTRID field: See the module configuration for the device specific reset value.

#### QuadSPI\_BUFB3CR field descriptions

Field	Description
0 ALLMST	All master enable. When set, buffer3 acts as an all-master buffer. Any AHB access with a master port number not matching with the master ID of buffer0 or buffer1 or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
1–15 Reserved	This field is reserved. Reserved.
16–23 ADATSZ	AHB data transfer size  Defines the data transfer size in 8 Bytes of an AHB triggered access to serial flash. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
24–27 Reserved	This field is reserved.

Table continues on the next page...

**QuadSPI\_BUF3CR field descriptions (continued)**

Field	Description
28–31 MSTRID	Master ID. The ID of the AHB master associated with BUFFER3. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

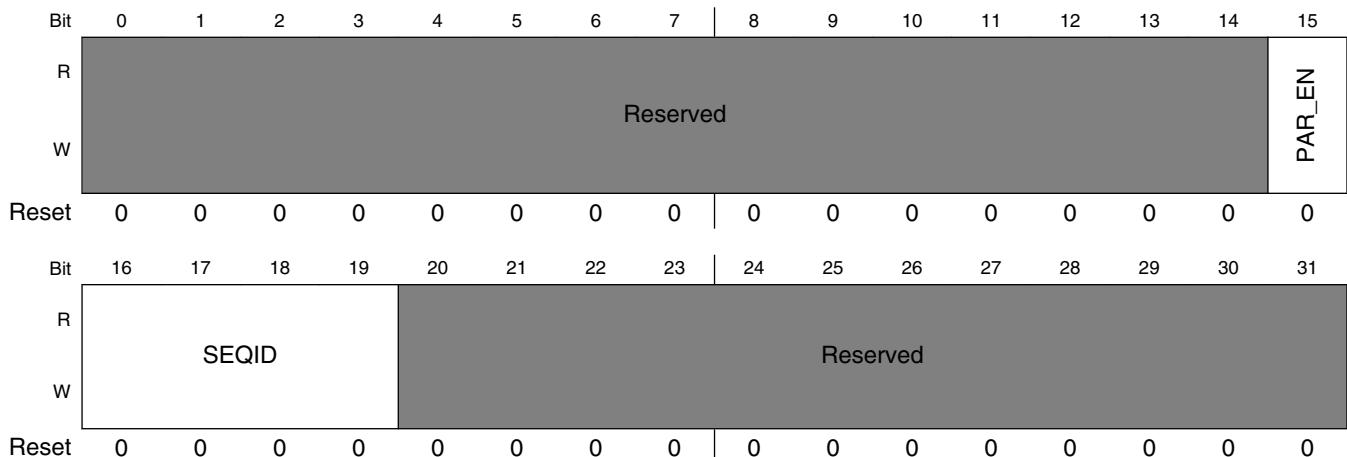
**30.4.2.8 Buffer Generic Configuration Register (QuadSPI\_BFGENCR)**

This register provides the generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field. If the PAR\_EN field is set, all the buffer accesses result in parallel accesses to the flashes.

*Write:*

- *QSPI\_SR[AHB\_ACC] = 0*

Address: 155\_0000h base + 20h offset = 155\_0020h

**QuadSPI\_BFGENCR field descriptions**

Field	Description
0–14 Reserved	This field is reserved.
15 PAR_EN	When set, a transaction to two serial flash devices is triggered in parallel mode. Refer to <a href="#">Parallel Flash Mode</a> for more details.
16–19 SEQID	Points to a sequence in the Look-up-table. The SEQID defines the bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. Refer to <a href="#">Look-up Table</a> .  <b>NOTE:</b> If the sequence pointer differs between the new and previous sequence then the user should reset this. See QSPI_SPTRCLR for more information.
20–31 Reserved	This field is reserved.

### 30.4.2.9 Buffer0 Top Index Register (QuadSPI\_BUFOIND)

This register specifies the top index of buffer0, which defines its size. Note that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned, as each buffer entry is 64 bits long.

The register value should be set to the desired number of bytes less 8. For example, setting BUFOIND to 0 gives 8 bytes, 1 gives 16 bytes etc.

The size of buffer0 is the difference between BUFOIND+8 and 0.

It is the responsibility of the software to ensure that BUFOIND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 30h offset = 155\_0030h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R																	TPINDX0																Reserved
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**QuadSPI\_BUFOIND field descriptions**

Field	Description
0–28 TPINDX0	Top index of buffer 0.
29–31 Reserved	This field is reserved. Reserved.

### 30.4.2.10 Buffer1 Top Index Register (QuadSPI\_BUFIIND)

This register specifies the top index of buffer1, which defines its size. Note that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned as each buffer entry is 64 bits long.

The size of buffer1 is the difference between BUFIIND and BUFOIND. The register value should be entered in bytes. For example, If BUFOIND = 0x100 then setting BUFIIND = 0x130 will set buffer1 size to 0x30 bytes.

It is the responsibility of the software to ensure that BUFIIND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

## Memory Map and Register Definition

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 34h offset = 155\_0034h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	TPINDEX1															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### QuadSPI\_BUFIIND field descriptions

Field	Description
0–28 TPINDEX1	Top index of buffer 1.
29–31 Reserved	This field is reserved.

## 30.4.2.11 Buffer2 Top Index Register (QuadSPI\_BUFIIND)

This register specifies the top index of buffer2, which defines its size. Note that that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned as each buffer entry is 64 bits long.

The size of buffer2 is the difference between the BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 will set buffer2 size to 0x50 bytes.

It is the responsibility of the software to ensure that BUF2IND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

*Write:*

- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 38h offset = 155\_0038h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																	TPINDEX2															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### QuadSPI\_BUFIIND field descriptions

Field	Description
0–28 TPINDEX2	Top index of buffer 2.
29–31 Reserved	This field is reserved.

### 30.4.2.12 Serial Flash Address Register (QuadSPI\_SFAR)

The module automatically translates this address on the memory map to the address on the flash itself. When operating in 24-bit mode, only bits 23-0 are sent to the flash. In 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 Refer to [Table 30-15](#) for the mapping between the access mode and the QSPI\_SFAR content and to [Normal Mode](#) for details about the command triggering and command execution. The software should ensure that the serial flash address provided in the QSPI\_SFAR register lies in the valid flash address range as defined in [Table 30-15](#).

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$

Address: 155\_0000h base + 100h offset = 155\_0100h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0

#### QuadSPI\_SFAR field descriptions

Field	Description
0–31 SFADR	Serial Flash Address. The register content is used as byte address for all following IP Commands.

### 30.4.2.13 Sampling Register (QuadSPI\_SMPR)

The Sampling Register allows configuration of how the incoming data from the external serial flash devices are sampled in the QuadSPI module.

*Write: Disabled Mode*

Address: 155\_0000h base + 108h offset = 155\_0108h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																Reserved

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

## Memory Map and Register Definition

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									0	FSDLY	FSPHS		0	HSDLY	HSPHS	HSENA
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### QuadSPI\_SMPR field descriptions

Field	Description
0–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13–15 Reserved	This field is reserved.
16–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 FSDLY	Full Speed Delay selection for SDR instructions. Select the delay with respect to the reference edge for the sample point valid for full speed commands.  0 One clock cycle delay 1 Two clock cycles delay.
26 FSPHS	Full Speed Phase selection for SDR instructions.  Select the edge of the sampling clock valid for full speed commands.  0 Select sampling at non-inverted clock 1 Select sampling at inverted clock.
27–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
29 HSDLY	Half Speed Delay selection for SDR instructions.  Only relevant when HSENA bit is set. Select the delay with respect to the reference edge for the sample point valid for half speed commands.  0 One clock cycle delay 1 Two clock cycle delay
30 HSPHS	Half Speed Phase selection for SDR instructions.  Only relevant when HSENA bit is set. Select the delay with respect to the reference edge for the sample point valid for half speed commands.  0 Select sampling at non-inverted clock 1 Select sampling at inverted clock
31 HSENA	Half Speed serial flash clock Enable  This bit enables the divide by 2 of the clock to the external serial flash device for all commands, only in SDR. Refer to <a href="#">Serial Flash Clock Frequency Limitations</a> for details.  0 Disable divide by 2 of serial flash clock for half speed commands 1 Enable divide by 2 of serial flash clock for half speed commands

### 30.4.2.14 RX Buffer Status Register (QuadSPI\_RBSR)

This register contains information related to the receive data buffer.

Address: 155\_0000h base + 10Ch offset = 155\_010Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RDCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved		RDBFL					Reserved								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**QuadSPI\_RBSR field descriptions**

Field	Description
0–15 RDCTR	Read Counter. Indicates how many entries of 4 bytes have been removed from the RX Buffer. For example, a value of 0x2 would indicate 8 bytes have been removed.  It is incremented by the number (QSPI_RBCT[WMRK] + 1) on RX Buffer POP event. The RX Buffer can be popped using DMA or pop flag QSPI_FR[RBDL]. The QSPI_RSER[RBDDE] defines which pop has to be done. For further details, refer to <a href="#">AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB31)</a> and "Data Transfer from the QuadSPI Module Internal Buffers" section in <a href="#">Flash Read</a> .
16–17 Reserved	This field is reserved.
18–23 RDBFL	RX Buffer Fill Level. Indicates how many entries of 4 bytes are still available in the RX Buffer. For example, a value of 0x2 would indicate 8 bytes are available.
24–31 Reserved	This field is reserved.

### 30.4.2.15 RX Buffer Control Register (QuadSPI\_RBCT)

This register contains control data related to the receive data buffer.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$

## Memory Map and Register Definition

Address: 155\_0000h base + 110h offset = 155\_0110h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved				RXBRD	Reserved			WMRK				
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### QuadSPI\_RBCT field descriptions

Field	Description
0–22 Reserved	This field is reserved.
23 RXBRD	RX Buffer Readout. This field specifies the access scheme for the RX Buffer readout. 0 RX Buffer content is read using the AHB Bus registers QSPI_ARDB0 to QSPI_ARDB31. For details, refer to <a href="#">Exclusive Access to Serial Flash for AHB Commands</a> . 1 RX Buffer content is read using the IP Bus registers QSPI_RBDR0 to QSPI_RBDR31.
24–26 Reserved	This field is reserved.
27–31 WMRK	RX Buffer Watermark. This field determines when the readout action of the RX Buffer is triggered. When the number of valid entries in the RX Buffer is equal to or greater than the number given by (WMRK+1) the QSPI_SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 would set the watermark to 4 bytes, 1 to 8 bytes, 2 to 12 bytes, and so on. For details, refer to <a href="#">DMA Usage</a> .

### 30.4.2.16 TX Buffer Status Register (QuadSPI\_TBSR)

This register contains information related to the transmit data buffer.

Address: 155\_0000h base + 150h offset = 155\_0150h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									TRCTR				Reserved	TRBFL			Reserved					Reserved										
W																	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

## QuadSPI\_TBSR field descriptions

Field	Description
0–15 TRCTR	Transmit Counter. This field indicates how many entries of 4 bytes have been written into the TX Buffer by host accesses. It is reset to 0 when a 1 is written to QSPI_MCR[CLR_TXF]. It is incremented on each write access to the QSPI_TBDR register when another word has been pushed onto the TX Buffer. When it is not cleared the TRCTR field wraps around to 0. Refer to <a href="#">TX Buffer Data Register (QuadSPI_TBDR)</a> for details.
16–18 Reserved	This field is reserved.
19–23 TRBFL	TX Buffer Fill Level. The TRBFL field contains the number of entries of 4 bytes each available in the TX Buffer for the QuadSPI module to transmit to the serial flash device.
24–31 Reserved	This field is reserved.

### 30.4.2.17 TX Buffer Data Register (QuadSPI TBDR)

The QSPI\_TBDR register provides access to the circular TX Buffer of depth 64 bytes . This buffer provides the data written into it as write data for the page programming commands to the serial flash device. Refer to [Table 30-24](#) for the byte ordering scheme. A write transaction on the flash with data size of less than 32 bits will lead to the removal of one data entry from the TX buffer. The valid bits will be used and the rest of the bits will be discarded.

*Write:*

- OSPI SR[*TXFULL*] = 0

*32-bit write access required*

Address: 155 0000h base + 154h offset = 155 0154h

## QuadSPI TBDR field descriptions

Field	Description
0-31 <b>TXDATA</b>	<p>TX Data</p> <p>On write access the data is written into the next available entry of the TX Buffer and the QPSI_TBSR[TRBFL] field is updated accordingly.</p> <p>On a read access, the last data written to the register is returned.</p>

### 30.4.2.18 Status Register (QuadSPI\_SR)

The QSPI\_SR register provides all available status information about SFM command execution and arbitration, the RX Buffer, TX Buffer, and the AHB Buffer.

Address: 155\_0000h base + 15Ch offset = 155\_015Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	Reserved		TXFULL	Reserved	Reserved		TXNE	RXDMA	Reserved		RXFULL	Reserved		RXWE	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	AHB3FUL	AHB2FUL	AHB1FUL	AHBOFUL	AHB3NE	AHB2NE	AHB1NE	AHBONE	AHBTRN	AHBGNT	Reserved	Reserved	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

**QuadSPI\_SR field descriptions**

Field	Description
0–2 Reserved	This field is reserved.
3 Reserved	This field is reserved.
4 TXFULL	TX Buffer Full. Asserted when no more data can be stored.
5 Reserved	This field is reserved.
6 Reserved	This field is reserved.
7 TXNE	TX Buffer Not Empty: Asserted when TX Buffer contains data.
8 RXDMA	RX Buffer DMA. Asserted when RX Buffer read out via DMA is active i.e DMA is requested or running.
9–11 Reserved	This field is reserved.
12 RXFULL	RX Buffer Full. Asserted when the RX Buffer is full, i.e. that QSPI_RBSR[RDBFL] field is equal to 32.

*Table continues on the next page...*

### QuadSPI\_SR field descriptions (continued)

Field	Description
13–14 Reserved	This field is reserved.
15 RXWE	RX Buffer Watermark Exceeded. Asserted when the number of valid entries in the RX Buffer exceeds the number given in the QSPI_RBCT[WMRK] field.
16 Reserved	This field is reserved.
17 AHB3FUL	AHB 3 Buffer Full. Asserted when AHB 3 buffer is full.
18 AHB2FUL	AHB 2 Buffer Full. Asserted when AHB 2 buffer is full.
19 AHB1FUL	AHB 1 Buffer Full. Asserted when AHB 1 buffer is full.
20 AHB0FUL	AHB 0 Buffer Full. Asserted when AHB 0 buffer is full.
21 AHB3NE	AHB 3 Buffer Not Empty. Asserted when AHB 3 buffer contains data.
22 AHB2NE	AHB 2 Buffer Not Empty. Asserted when AHB 2 buffer contains data.
23 AHB1NE	AHB 1 Buffer Not Empty. Asserted when AHB 1 buffer contains data.
24 AHB0NE	AHB 0 Buffer Not Empty. Asserted when AHB 0 buffer contains data.
25 AHBTRN	AHB Access Transaction pending. Asserted when there is a pending request on the AHB interface. Refer to the AMBA specification for details.
26 AHBGNT	AHB Command priority Granted: Asserted when another module has been granted priority of AHB Commands against IP Commands. For details refer to <a href="#">Command Arbitration</a> .
27 Reserved	This field is reserved.
28 RESERVED	This field is reserved.
29 AHB_ACC	AHB Access. Asserted when the transaction currently executed was initiated by AHB bus.
30 IP_ACC	IP Access. Asserted when transaction currently executed was initiated by IP bus.
31 BUSY	Module Busy. Asserted when module is currently busy handling a transaction to an external flash device.

### 30.4.2.19 Flag Register (QuadSPI\_FR )

The QSPI\_FR register provides all available flags about SFM command execution and arbitration which may serve as source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash device itself but only to the behavior and conditions visible in the QuadSPI module.

*Write: Enabled Mode*

Address: 155\_0000h base + 160h offset = 155\_0160h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	Reserved	Reserved	Reserved	TBFF	TBUF	Reserved	ILLINE	Reserved	Reserved	Reserved	Reserved	Reserved	RBDF	RBDF	
W	w1c	w1c	w1c	w1c				w1c					w1c	w1c		
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ABSEF	Reserved	Reserved	ABOF	IUEF	Reserved	IPAEF	IPIEF	Reserved	IPGEF	Reserved	Reserved	Reserved	TFF		
W	w1c			w1c	w1c		w1c	w1c		w1c				w1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**QuadSPI\_FR field descriptions**

Field	Description
0 Reserved	This field is reserved.
1–2 RESERVED	This field is reserved.

Table continues on the next page...

## QuadSPI\_FR field descriptions (continued)

Field	Description
3 Reserved	This field is reserved.
4 TBFF	TX Buffer Fill Flag. Before writing to the TX buffer, this bit should be cleared. Then this bit has to be read back. If the bit is set, the TX Buffer can take more data. If the bit remains cleared, the TX buffer is full. Refer to <a href="#">Tx Buffer Operation</a> for details.
5 TBUF	TX Buffer Underrun Flag. Set when the module tried to pull data although TX Buffer was empty. The IP Command leading to the TX Buffer underrun is continued (data sent to the serial flash device is undefined). The application must clear the TX Buffer in response to this event by writing a 1 to the QSPI_MCR[CLR_TXF] bit.
6–7 Reserved	This field is reserved.
8 ILLINE	Illegal Instruction Error Flag. Set when an illegal instruction is encountered by the controller in any of the sequences. Refer to <a href="#">Table 30-22</a> for a list of legal instructions.
9–13 Reserved	This field is reserved.
14 RBOF	RX Buffer Overflow Flag. Set when not all the data read from the serial flash device could be pushed into the RX Buffer.  The IP Command leading to this condition is continued until the number of bytes according to the QSPI_IPCR[IDATSZ] field has been read from the serial flash device.  The content of the RX Buffer is not changed.
15 RBDF	RX Buffer Drain Flag. Will be set if the QuadSPI_SR[RXWE] status bit is asserted.  Writing 1 into this bit triggers one of the following actions: <ul style="list-style-type: none"> <li>If the RX Buffer has up to QuadSPI_RBCT[WMRK] valid entries then the flag is cleared.</li> <li>If the RX Buffer has more than QuadSPI_RBCT[WMRK] valid entries and the QuadSPI_RSER[RBDDE] bit is not set (flag driven mode) a RX Buffer POP event is triggered.</li> </ul> The flag remains set if the RX Buffer contains more than QuadSPI_RBCT[WMRK] valid entries after the RX Buffer POP event is finished.  The flag is cleared if the RX Buffer contains less than or equal to QuadSPI_RBCT[WMRK] valid entries after the RX Buffer POP event is finished.  Refer to "Receive Buffer Drain Interrupt or DMA Request" section in <a href="#">Normal Mode Interrupt and DMA Requests</a> , for details.
16 ABSEF	AHB Sequence Error Flag. Set when the execution of an AHB Command is started with a WRITE Command in the sequence pointed to by the QSPI_BUFXCR register. (QSPI_BUFXCR implies any one of QSPI_BUF0CR/QSPI_BUF1CR/QSPI_BUF2CR/QSPI_BUF3CR.)  Communication with the serial flash device is terminated before the execution of WRITE command by the QuadSPI module.  The AHB bus request which triggered this command is answered with an ERROR response.
17 Reserved	Reserved  This field is reserved.
18 Reserved	Reserved  This field is reserved.
19 ABOF	AHB Buffer Overflow Flag. Set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if the QSPI_BUFXCR[ADATSZ] field is programmed incorrectly.

*Table continues on the next page...*

**QuadSPI\_FR field descriptions (continued)**

Field	Description
	The AHB Command leading to this condition is continued until the number of entries according to the QSPI_BUFXCR[ADATSZ] field has been read from the serial flash device. The content of the AHB Buffer is not changed.
20 IUEF	IP Command Usage Error Flag. Set when in parallel flash mode the execution of an IP Command is started and the sequence pointed to by the sequence ID contains a WRITE command. Refer to <a href="#">Table 30-22</a> table for the related commands. Communication with the serial flash device is terminated before the execution of WRITE command by the QuadSPI module.
21–23 Reserved	This field is reserved.
24 IPAEF	IP Command Trigger during AHB Access Error Flag. Set when the following condition occurs: <ul style="list-style-type: none"> <li>A write access occurs to the QSPI_IPCR[SEQID] field and the QSPI_SR[AHB_ACC] bit is set. Any command leading to the assertion of the IPAEF flag is ignored.</li> </ul>
25 IPIEF	IP Command Trigger could not be executed Error Flag. Set when the QSPI_SR[IP_ACC] bit is set (i.e. an IP triggered command is currently executing) and any of the following conditions occurs: <ul style="list-style-type: none"> <li>Write access to the QSPI_IPCR register. Any command leading to the assertion of the IPIEF flag is ignored</li> <li>Write access to the QSPI_SFAR register.</li> <li>Write access to the QSPI_RBCT register.</li> </ul>
26 Reserved	This field is reserved.
27 IPGEF	IP Command Trigger during AHB Grant Error Flag. Set when the following condition occurs: <ul style="list-style-type: none"> <li>A write access occurs to the QSPI_IPCR[SEQID] field and the QSPI_SR[AHBGNT] bit is set. Any command leading to the assertion of the IPGEF flag is ignored.</li> </ul>
28–30 Reserved	This field is reserved.
31 TFF	IP Command Transaction Finished Flag. Set when the QuadSPI module has finished a running IP Command. If an error occurred the related error flags are valid, at the latest, in the same clock cycle when the TFF flag is asserted.

**30.4.2.20 Interrupt and DMA Request Select and Enable Register (QuadSPI\_RSER)**

The QuadSPI\_RSER register provides enables and selectors for the interrupts in the QuadSPI module.

**NOTE**

Each flag of the QuadSPI\_FR register enabled as source for an interrupt prevents the QuadSPI module from entering Stop Mode or Module Disable Mode when this flag is set.

*Write: Anytime*

## Memory Map and Register Definition

Address: 155\_0000h base + 164h offset = 155\_0164h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	Reserved	Reserved	TBFIE	TBUIE	Reserved	Reserved	ILLINE	Reserved	Reserved	RBDDE	Reserved	Reserved	Reserved	RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	ABSEIE	Reserved	Reserved	ABOIE	IUEIE	Reserved	IPAEIE	IPIEIE	Reserved	IPGEIE	Reserved	Reserved	TFIE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### QuadSPI\_RSER field descriptions

Field	Description
0 Reserved	Reserved This field is reserved.
1–2 Reserved	This field is reserved.
3 RESERVED	This field is reserved.
4 TBFIE	TX Buffer Fill Interrupt Enable 0 No TBFF interrupt will be generated 1 TBFF interrupt will be generated
5 TBUIE	TX Buffer Underrun Interrupt Enable 0 No TBUF interrupt will be generated 1 TBUF interrupt will be generated
6 Reserved	This field is reserved.
7 Reserved	This field is reserved.
8 ILLINIE	Illegal Instruction Error Interrupt Enable. Triggered by ILLINE flag in QSPI_FR 0 No ILLINE interrupt will be generated 1 ILLINE interrupt will be generated

Table continues on the next page...

**QuadSPI\_RSER field descriptions (continued)**

Field	Description
9 Reserved	This field is reserved.
10 RBDDE	RX Buffer Drain DMA Enable: Enables generation of DMA requests for RX Buffer Drain. When this bit is set DMA requests are generated as long as the QSPI_SR[RXWE] status bit is set.  0 No DMA request will be generated 1 DMA request will be generated
11–13 Reserved	This field is reserved.
14 RBOIE	RX Buffer Overflow Interrupt Enable  0 No RBOF interrupt will be generated 1 RBOF interrupt will be generated
15 RBDIE	RX Buffer Drain Interrupt Enable: Enables generation of IRQ requests for RX Buffer Drain. When this bit is set the interrupt is asserted as long as the QuadSPI_SR[RBDF] flag is set.  0 No RBDF interrupt will be generated 1 RBDF Interrupt will be generated
16 ABSEIE	AHB Sequence Error Interrupt Enable: Triggered by ABSEF flags of QSPI_FR  0 No ABSEF interrupt will be generated 1 ABSEF interrupt will be generated
17 Reserved	Reserved  This field is reserved.
18 Reserved	Reserved  This field is reserved.
19 ABOIE	AHB Buffer Overflow Interrupt Enable  0 No ABOF interrupt will be generated 1 ABOF interrupt will be generated
20 IUEIE	IP Command Usage Error Interrupt Enable  0 No IUEF interrupt will be generated 1 IUEF interrupt will be generated
21–23 Reserved	This field is reserved.
24 IPAEIE	IP Command Trigger during AHB Access Error Interrupt Enable  0 No IPAEF interrupt will be generated 1 IPAEF interrupt will be generated
25 IPIEIE	IP Command Trigger during IP Access Error Interrupt Enable  0 No IPIEF interrupt will be generated 1 IPIEF interrupt will be generated
26 Reserved	This field is reserved.

*Table continues on the next page...*

**QuadSPI\_RSER field descriptions (continued)**

Field	Description
27 IPGEIE	IP Command Trigger during AHB Grant Error Interrupt Enable  0 No IPGEF interrupt will be generated 1 IPGEF interrupt will be generated
28–30 Reserved	This field is reserved. Reserved.
31 TFIE	Transaction Finished Interrupt Enable  0 No TFF interrupt will be generated 1 TFF interrupt will be generated

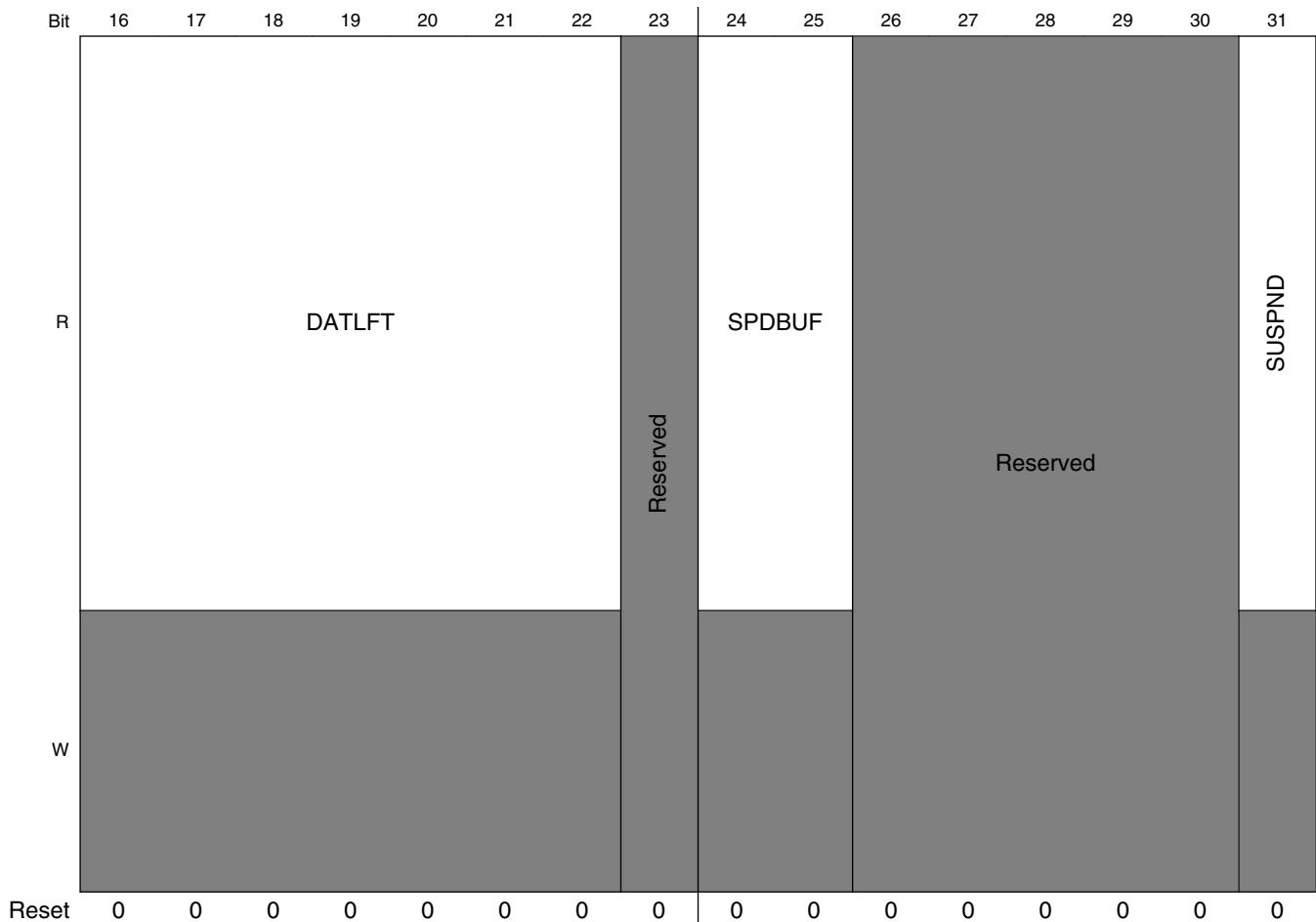
### 30.4.2.21 Sequence Suspend Status Register (QuadSPI\_SPNDST)

The sequence suspend status register provides information specific to any suspended sequence. An AHB sequence may be suspended when a high priority AHB master makes an access before the AHB sequence completes the data transfer requested.

Address: 155\_0000h base + 168h offset = 155\_0168h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## Memory Map and Register Definition



### QuadSPI\_SPNDST field descriptions

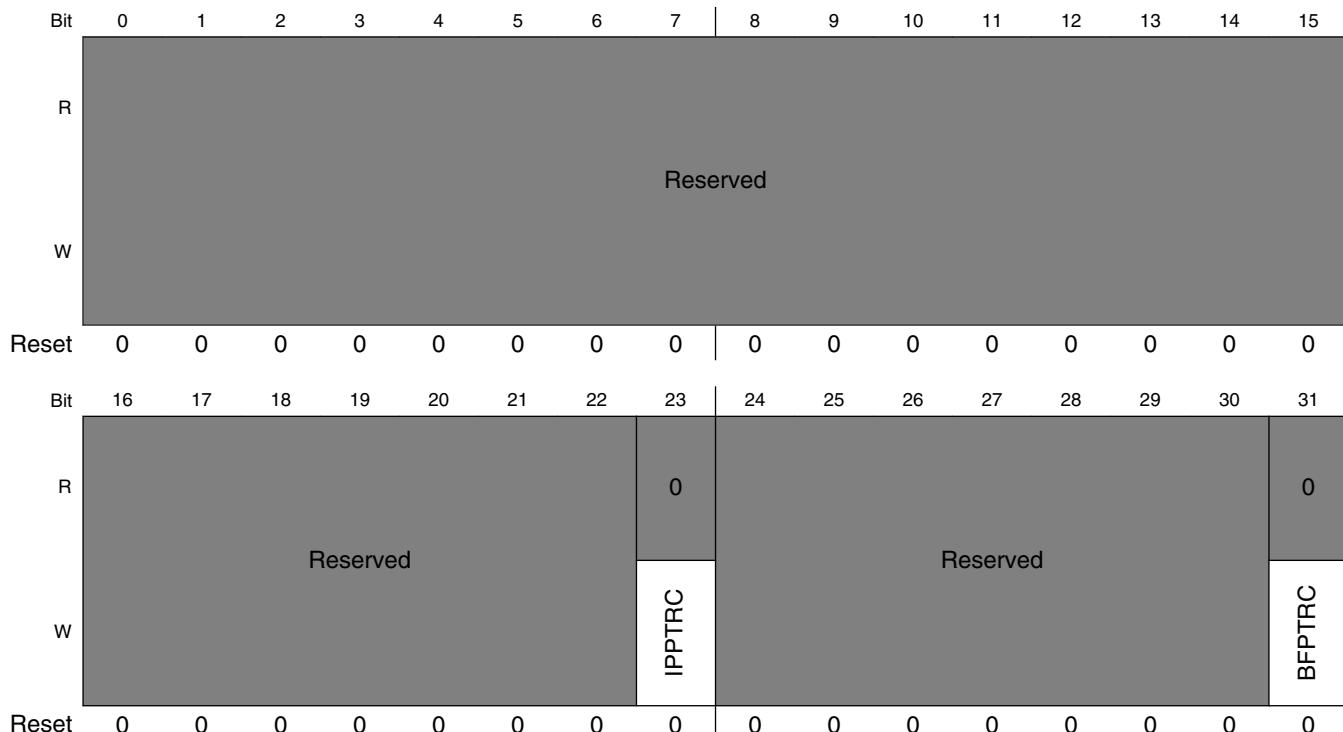
Field	Description
0–15 Reserved	This field is reserved.
16–22 DATLFT	Data left: Provides information about the amount of data left to be read in the suspended sequence. Valid only when SUSPND is set to 1'b1. Value in terms of 64 bits or 8 bytes
23 Reserved	This field is reserved.
24–25 SPDBUF	Suspended Buffer: Provides the suspended buffer number. Valid only when SUSPND is set to 1'b1
26–30 Reserved	This field is reserved.
31 SUSPND	When set, it signifies that a sequence is in suspended state

### 30.4.2.22 Sequence Pointer Clear Register (QuadSPI\_SPTRCLR)

The sequence pointer clear register provides bits to reset the IP and Buffer sequence pointers. The sequence pointer contains the index of which instruction within the LUT entry is to be executed next. For example, if the LUT entry ends on a JMP\_ON\_CS value of 2, the index will be stored as 2.

The software should reset the sequence pointers whenever the sequence ID is changed by updating the SEQID field in QSPI\_IPCR or QSPI\_BFGENCR.

Address: 155\_0000h base + 16Ch offset = 155\_016Ch



**QuadSPI\_SPTRCLR field descriptions**

Field	Description
0–22 Reserved	This field is reserved.
23 IPPTRC	IP Pointer Clear: 1: Clears the sequence pointer for IP accesses as defined in QuadSPI_IPCR This is a self-clearing field.
24–30 Reserved	This field is reserved. Reserved.
31 BFPTRC	Buffer Pointer Clear: 1: Clears the sequence pointer for AHB accesses as defined in QuadSPI_BFGENCR. This is a self-clearing field.

### **30.4.2.23 Serial Flash A1 Top Address (QuadSPI\_SFA1AD)**

The QSPI\_SFA1AD register provides the address mapping for the serial flash A1. The difference between QSPI\_SFA1AD[TPADA1] and QSPI\_AMBA\_BASE defines the size of the memory map for serial flash A1.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$
  - $QSPI\_SR[AHB\_ACC] = 0$

Address: 155 0000h base + 180h offset = 155 0180h

\* Notes:

- TPADA1 field: See the module configuration for the device specific reset values.

## QuadSPI SFA1AD field descriptions

Field	Description
0–21 TPADA1	Top address for Serial Flash A1. In effect, TPADxx is the first location of the next memory.
22–31 Reserved	This field is reserved.

### **30.4.2.24 Serial Flash A2 Top Address (QuadSPI\_SFA2AD)**

The QSPI\_SFA2AD register provides the address mapping for the serial flash A2. The difference between QSPI\_SFA2AD[TPADA2] and QSPI\_SFA1AD[TPADA1] defines the size of the memory map for serial flash A2.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$
  - $OSPI\_SR[AHB\_ACC] = 0$

Address: 155 0000h base + 184h offset = 155 0184h

### \* Notes:

- TPADA2 field: See the module configuration for the device specific reset values.

### QuadSPI\_SFA2AD field descriptions

Field	Description
0–21 TPADA2	Top address for Serial Flash A2. In effect, TPxxAD is the first location of the next memory.
22–31 Reserved	This field is reserved.

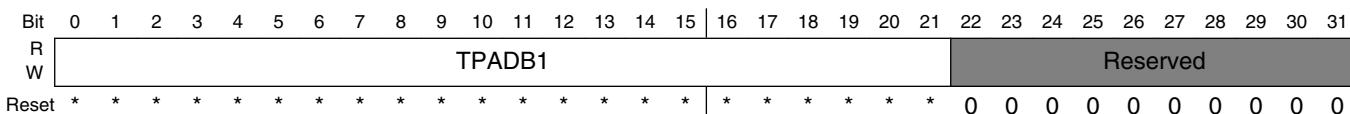
### 30.4.2.25 Serial Flash B1Top Address (QuadSPI\_SFB1AD)

The QSPI\_SFB1AD register provides the address mapping for the serial flash B1. The difference between QSPI\_SFB1AD[TPADB1] and QSPI\_SFA2AD[TPADA2] defines the size of the memory map for serial flash B1.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$
- $QSPI\_SR[AHB\_ACC] = 0$

Address: 155\_0000h base + 188h offset = 155\_0188h



\* Notes:

- TPADB1 field: See the module configuration for the device specific reset values.

### QuadSPI\_SFB1AD field descriptions

Field	Description
0–21 TPADB1	Top address for Serial Flash B1. In effect, TPxxAD is the first location of the next memory.
22–31 Reserved	This field is reserved.

### 30.4.2.26 Serial Flash B2Top Address (QuadSPI\_SFB2AD)

The QSPI\_SFB2AD register provides the address mapping for the serial flash B2. The difference between QSPI\_SFB2AD[TPADB2] and QSPI\_SFB1AD[TPADB1] defines the size of the memory map for serial flash B2.

*Write:*

- $QSPI\_SR[IP\_ACC] = 0$
- $QSPI\_SR[AHB\_ACC] = 0$

## Memory Map and Register Definition

Address: 155\_0000h base + 18Ch offset = 155\_018Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

\* Notes:

- TPADB2 field: See the module configuration for the device specific reset values.

## QuadSPI\_SFB2AD field descriptions

Field	Description
0–21 TPADB2	Top address for Serial Flash B2. In effect, TPxxAD is the first location of the next memory.
22–31 Reserved	This field is reserved.

## 30.4.2.27 RX Buffer Data Register (QuadSPI\_RBDRn)

The QuadSPI\_RBDR registers provide access to the individual entries in the RX Buffer. Refer to [Table 30-24](#) for the byte ordering scheme.

QuadSPI\_RBDR0 corresponds to the actual position of the read pointer within the RX Buffer. The number of valid entries available depends from the number of RX Buffer entries implemented and from the number of valid buffer entries available in the RX Buffer.

Example 1, RX Buffer filled completely with 32 words: In this case the address range for valid read access extends from QuadSPI\_RBDR0 to QuadSPI\_RBDR31.

Example 2, RX Buffer filled with 5 valid words: RX Buffer fill level QuadSPI\_RBSR[RDBFL] is 5. In this case an access to QuadSPI\_RBDR4 provides the last valid entry.

Any access beyond the range of valid RX Buffer entries provides undefined results.

Address: 155\_0000h base + 200h offset + (4d × i), where i=0d to 31d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

## QuadSPI\_RBDRn field descriptions

Field	Description
0–31 RXDATA	RX Data. The RXDATA field contains the data associated with the related RX Buffer entry. Data format and byte ordering is given in <a href="#">Byte Ordering of Serial Flash Read Data</a> .

### 30.4.2.28 LUT Key Register (QuadSPI\_LUTKEY)

The LUT Key register contains the key to lock and unlock the Look-up-table. Refer to [Look-up Table](#) for details.

*Write: Anytime*

Address: 155\_0000h base + 300h offset = 155\_0300h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 1 0 1 1 0 1 0 1 1 1 0 0 0 0 | 0 1 0 1 1 0 1 0 1 1 1 0 0 0 0 0

#### QuadSPI\_LUTKEY field descriptions

Field	Description
0–31 KEY	The key to lock or unlock the LUT. The KEY is 0x5AF05AF0. The read value is always 0x5AF05AF0

### 30.4.2.29 LUT Lock Configuration Register (QuadSPI\_LCKCR)

The LUT lock configuration register is used along with QSPI\_LUTKEY register to lock or unlock the LUT. This register has to be written immediately after QSPI\_LUTKEY register for the lock or unlock operation to be successful. Refer to [Look-up Table](#) for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

*Write: Just after writing the LUT Key Register*

(QSPI\_LUTKEY)

Address: 155\_0000h base + 304h offset = 155\_0304h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																

Reset 0 0 0 0 0 0 0 0 | 0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																

Reset 0 0 0 0 0 0 0 0 | 0

### QuadSPI\_LCKCR field descriptions

Field	Description
0–29 Reserved	This field is reserved.
30 UNLOCK	Unlocks the LUT when the following two conditions are met: 1. This register is written just after the <a href="#">LUT Key Register (QuadSPI_LUTKEY)</a> 2. The LUT key register was written with 0x5AF05AF0 key
31 LOCK	Locks the LUT when the following condition is met: 1. This register is written just after the <a href="#">LUT Key Register (QuadSPI_LUTKEY)</a> 2. The LUT key register was written with 0x5AF05AF0 key

### 30.4.2.30 Look-up Table register (QuadSPI\_LUTn)

The LUT registers are a look-up-table for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash transaction. There are a total of 64 LUT registers. These 64 registers are divided into groups of 4 registers that make a valid sequence. Therefore, QSPI\_LUT[0], QSPI\_LUT[4], QSPI\_LUT[8] ..... QSPI\_LUT[60] are the starting registers of a valid sequence. Each of these sets of 4 registers can have a maximum of 8 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT63. A maximum of 16 sequences can be defined at one time. [Look-up Table](#) describes the LUT registers in detail.

#### NOTE

The reset values for LUT0 and LUT1 are 0818\_0403h and 2400\_1C08h, respectively.

*Write: Once the LUT is unlocked*

Address: 155\_0000h base + 310h offset + (4d × i), where i=0d to 63d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	INSTR1							PAD1	OPRND1							
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	INSTR0							PAD0	OPRND0							
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	

\* Notes:

- The reset values for LUT0 and LUT1 are 0818\_0403h and 2400\_1C08h respectively.

### QuadSPI\_LUTn field descriptions

Field	Description
0–5 INSTR1	Instruction 1
6–7 PAD1	Pad information for INSTR1. 00 1 Pad 01 2 Pads 10 4 Pads 11 NA
8–15 OPRND1	Operand for INSTR1.
16–21 INSTR0	Instruction 0
22–23 PAD0	Pad information for INSTR0. 00 1 Pad 01 2 Pads 10 4 Pads 11 NA
24–31 OPRND0	Operand for INSTR0.

### 30.4.3 Serial Flash Address Assignment

The serial flash address assignment may be modified by writing into [Serial Flash A1 Top Address \(QuadSPI\\_SFA1AD\)](#) and [Serial Flash A2 Top Address \(QuadSPI\\_SFA2AD\)](#) for device A and into [Serial Flash B1Top Address \(QuadSPI\\_SFB1AD\)](#) and [Serial Flash B2Top Address \(QuadSPI\\_SFB2AD\)](#) for device B. The following table shows how different access modes are related to the address specified for the next SFM Command. Note that this address assignment is valid for both IP and AHB commands.

**Table 30-15. Serial Flash Address Assignment**

Parameter	Function	Access Mode
QSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address	
TOP_ADDR_MEMA1(T_PADA1)	Top address for the external flash A1 (first device of the dual die flash A, or the first of the two independent flashes sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QSPI_AMBA_BASE will be routed to <b>Serial Flash A1</b>
TOP_ADDR_MEMA2(T_PADA2)	Top address for the external flash A2 (second device of the dual die flash A, or the second of the two independent flashes sharing the IOFA).	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 will be routed to <b>Serial Flash A2</b>

*Table continues on the next page...*

**Table 30-15. Serial Flash Address Assignment (continued)**

Parameter	Function	Access Mode
TOP_ADDR_MEMB1(T PADB1)	Top address for the external flash B1 (first device of the dual die flash B, or the first of the two independent flashes sharing the IOFB)	Any access to the address space between TOP_ADDR_MEMB1 and TOP_ADDR_MEMA2 will be routed to <b>Serial Flash B1</b>
TOP_ADDR_MEMB2(T PADB2)	Top address for the external flash B2 (second device of the dual die flash B or the second of the two independent flashes sharing the IOFB)	Any access to the address space between TOP_ADDR_MEMB2 and TOP_ADDR_MEMB1 will be routed to <b>Serial Flash A2</b>

### 30.4.4 AMBA Bus Register Memory Map

QSPI\_AMBA\_BASE defines the address to be used as start address of the serial flash device as defined by the system memory map..

**Table 30-16. QuadSPI AMBA Bus Memory Map**

Address	Register Name
<a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A</a>	
QSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 0x01)	<a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A</a> Refer to <a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A</a> for details and to <a href="#">Table 30-24</a> and <a href="#">Table 30-28</a> for information about the byte ordering.
<a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B</a>	
TOP_ADDR_MEMA2 to (TOP_ADDR_MEMB2 - 0x01)	<a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B</a> Refer to <a href="#">Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B</a> for details and to <a href="#">Table 30-24</a> and <a href="#">Table 30-28</a> for information about the byte ordering.
<a href="#">Parallel Flash Mode</a>	
QSPI_AMBA_BASE to (TOP_ADDR_MEMB2 - 0x01)	<a href="#">Parallel Flash Mode</a> Refer to <a href="#">Parallel Flash Mode</a> for details and to <a href="#">Table 30-27</a> and <a href="#">Table 30-28</a> for information about the byte ordering.
<a href="#">AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB31)</a>	
QSPI_ARDB_BASE to... (32 * 4 Byte) QSPI_ARDB_BASE + 0x0000_01FF	<a href="#">AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB31)</a> Refer to <a href="#">Table 30-24</a> and <a href="#">Table 30-26</a> for information about the byte ordering.

### Note

Any read access to non-implemented addresses will provide undefined results.

In case single die flash devices, TOP\_ADDR\_MEMA2 and TOP\_ADDR\_MEMB2 should be initialized/programmed to TOP\_ADDR\_MEMA1 and TOP\_ADDR\_MEMB1 respectively- in effect, setting the size of these devices to 0. This would ensure that the complete memory map is assigned to only one flash device.

Parallel Flash Mode is valid only for commands related to data read from the serial flash. The first device of flash A has to be paired with the first device of flash B and the second device of flash A has to be paired with the second device of flash B in parallel mode. Parallel mode is selected via the QSPI\_BFGENCR[PAR\_EN] bit for all masters in AHB driven mode and via the QSPI\_IPCR[PAR\_EN] in IP driven mode. In parallel mode, the incoming address (SFAR address in case of IP initiated transactions and the incoming AHB address in case of AHB initiated transactions) is divided by 2 and sent to the two flashes connected in parallel.

Any IP Command other than data read in Parallel Flash Mode will result in the assertion of the QSPI\_FR[IUEF] flag and any AHB Command other than data read in Parallel Flash Mode will result in the assertion of the QSPI\_FR[ABSEF] flag.

In the Individual Flash Modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash address is determined by SFADR[8:31] or SFADR[0:31] as given in the table above.

In Parallel Flash Mode, both flashes are read with the same starting address of 3/4 (as programmed in the instruction/operand in the sequence) bytes in size. This address is derived from SFADR[7:30] or SFADR[0:30] as given in the table above. The LSB of the SFADR field is used to select the appropriate bits of both flash devices to combine the byte corresponding to the selected address.

### 30.4.5 AHB Bus Register Memory Map Descriptions

This chapter contains definitions of registers in the AMBA address space.

### 30.4.5.1 AHB Bus Access Considerations

It has to be noted that all logic in the QuadSPI module implementing the AHB Bus access is designed to read the content of an external serial flash device. Therefore the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- Any write access is answered with the ERROR condition according to the AMBA AHB Specification. No write occurs.
- Any AHB Command resulting in the assertion of the QSPI\_FR[ABSEF] flag is answered with the ERROR condition according to the AMBA\_AHB specification. The resulting AHB Command is ignored.
- AHB Bus access types fully supported are NONSEQ and BUSY.
- AHB access type SEQ is treated in the same way like NONSEQ. Refer to the AMBA AHB Specification for further details.

### 30.4.5.2 Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A

Starting with address QSPI\_AMBA\_BASE the content of the first external serial flash devices is mapped into the address space of the device containing the QuadSPI module. Serial flash address byte address 0x0 corresponds to bus address QSPI\_AMBA\_BASE with increasing order. Assuming that a dual-die flash is connected on the first set of external pads, the address space is divided into two parts, one for each device of the dual die package. Refer to the following table for the address mapping. The byte ordering for 32 bit access is given in [Table 30-24](#) and for 64 bit read access the byte ordering is given in [Table 30-28](#).

**Table 30-17. Memory Mapped Individual Flash Mode - Flash A Address Scheme**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash Byte Address	Flash Device
QSPI_AMBA_BASE + 0x00	QSPI_AMBA_BASE + 0x00	0x00_0000 to 0x00_0003	A1
QSPI_AMBA_BASE + 0x04		0x00_0004 to 0x00_0007	
...		...	
TOP_ADDR_MEMA1 - 0x08		(TOP_ADDR_MEMA1 - 0x08) to (TOP_ADDR_MEMA1 - 0x04 - 0x01)	
TOP_ADDR_MEMA1 - 0x04	TOP_ADDR_MEMA1 - 0x08	(TOP_ADDR_MEMA1 - 0x04) to (TOP_ADDR_MEMA1 - 0x01)	
TOP_ADDR_MEMA1 + 0x00	TOP_ADDR_MEMA1 + 0x00_0000	0x00_0000 to 0x00_0003	A2
TOP_ADDR_MEMA1 + 0x04		0x00_0004 to 0x00_0007	
.....	...	...	

*Table continues on the next page...*

**Table 30-17. Memory Mapped Individual Flash Mode - Flash A Address Scheme (continued)**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash Byte Address	Flash Device
TOP_ADDR_MEMA2 - 0x08	TOP_ADDR_MEMA2 - 0x08	(TOP_ADDR_MEMA2 - 0x08) to (TOP_ADDR_MEMA2 - 0x04 - 0x01)	
TOP_ADDR_MEMA2 - 0x04		(TOP_ADDR_MEMA2 - 0x04) to (TOP_ADDR_MEMA2 - 0x01)	

The available address range depends from the size of the external serial flash device. Any access beyond the size of the external serial flash provides undefined results.

For details concerning the read process refer to [Flash Read](#).

### 30.4.5.3 Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B

Starting with address TOP\_ADDR\_MEMA2 the content of the first external serial flash devices is mapped into the address space of the device containing the QuadSPI module. Serial flash address byte address 0x0 corresponds to bus address TOP\_ADDR\_MEMA2 with increasing order. Assuming that a dual-die flash is connected on the first set of external pads, the address space is divided into two parts, one for each device of the dual die package. Refer the following table for the address mapping. The byte ordering for 32 bit access is given in [Table 30-24](#) and for 64 bit read access the byte ordering is given in [Table 30-28](#).

**Table 30-18. Memory Mapped Individual Flash Mode - Flash B Address Scheme**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash Byte Address	Flash Device
TOP_ADDR_MEMA2 + 0x00	TOP_ADDR_MEMA2 + 0x00	0x00_0000 to 0x00_0003	B1
TOP_ADDR_MEMA2 + 0x04		0x00_0004 to 0x00_0007	
...		...	
TOP_ADDR_MEMB1 - 0x08		(TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0x08) to (TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0x04 - 0x01)	
TOP_ADDR_MEMB1 - 0x04	TOP_ADDR_MEMB1 - 0x08	(TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0x04) to (TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0x01)	B2
TOP_ADDR_MEMB1 + 0x00		0x00_0000 to 0x00_0003	
TOP_ADDR_MEMB1 + 0x04	TOP_ADDR_MEMB1 + 0x00_0000	0x00_0004 to 0x00_0007	

Table continues on the next page...

**Table 30-18. Memory Mapped Individual Flash Mode - Flash B Address Scheme (continued)**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash Byte Address	Flash Device
.....	...	...	
TOP_ADDR_MEMB2 - 0x08	TOP_ADDR_MEMA2 - 0x08	(TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x08) to (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x04 - 0x01)	
TOP_ADDR_MEMB2 - 0x04		(TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x04) to (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x01)	

The available address range depends from the size of the external serial flash device. Any access beyond the size of the external serial flash provides undefined results.

For details concerning the read process refer to [Flash Read](#).

### 30.4.5.4 Parallel Flash Mode

Any of the AHB flexible-buffers can be configured to work in parallel flash mode by programming the QSPI\_BFGENCR[PAR\_EN] bit to '1'. When parallel mode is set, Flash A1 is paired with Flash B1 and Flash A2 is paired with Flash B2. In parallel mode, software should ensure that the size of Flash A1(A2) is equal to the size of Flash B1(B2).

Reads from any even AHB bus address provides bits [7:4] of both serial flash devices and reads from any odd AHB bus address provides bits [3:0] of both flash devices. Refer to the following table for the address mapping. The byte ordering for 32 bit access is given in [Table 30-26](#) and for 64 bit read access the byte ordering is given in [Table 30-28](#).

**Table 30-19. Memory Mapped Parallel Flash Mode Address Scheme**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash A Byte Address	Serial Flash B Byte Address
QSPI_AMBA_BASE + 0x0000_0000	QSPI_AMBA_BASE + 0x00	0x00_0000	0x00_0000
		-	-
QSPI_AMBA_BASE + 0x0000_0004	For details, please refer to <a href="#">Parallel mode</a> and <a href="#">Dual Die Flashes</a> .	0x00_0001	0x00_0001
		0x00_0002	0x00_0002
QSPI_AMBA_BASE + 0x0000_0008	QSPI_AMBA_BASE + 0x08	-	-
		0x00_0003	0x00_0003
		0x00_0004	0x00_0004
		-	-

Table continues on the next page...

**Table 30-19. Memory Mapped Parallel Flash Mode Address Scheme (continued)**

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash A Byte Address	Serial Flash B Byte Address
		0x00_0005	0x00_0005
QSPI_AMBA_BASE + 0x0000_000C		0x00_0006	0x00_0006
		-	-
		0x00_0007	0x00_0007
...	...	...	...
TOP_ADDR_MEMB2 - 0x08		(TOP_ADDR_MEMB2 - QSPI_AMBA_BASE - 0x08)/2	(TOP_ADDR_MEMB2 - QSPI_AMBA_BASE - 0x08)/2
TOP_ADDR_MEMB2 - 0x04	TOP_ADDR_MEMB2 - 0x08	(TOP_ADDR_MEMB2 - QSPI_AMBA_BASE - 0x04)/2 + 0x01	(TOP_ADDR_MEMB2 - QSPI_AMBA_BASE - 0x04)/2 + 0x01

The available address range covers 27 address bits, corresponding to 128 MB per flash device. The usable space depends from the size of the external serial flash devices. Any access beyond the size of the external serial flash provides undefined results.

For details concerning the read process refer to [Flash Read](#).

### 30.4.5.5 AHB RX Data Buffer (QSPI\_ARDB0 to QSPI\_ARDB31)

#### NOTE

See the System Memory map in this document for the base address of the QSPI AHB RX Data Buffer.

#### memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	AHB RX Data Buffer register (ARDB0)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
4	AHB RX Data Buffer register (ARDB1)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
8	AHB RX Data Buffer register (ARDB2)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
C	AHB RX Data Buffer register (ARDB3)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
10	AHB RX Data Buffer register (ARDB4)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>

Table continues on the next page...

**memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
14	AHB RX Data Buffer register (ARDB5)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
18	AHB RX Data Buffer register (ARDB6)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
1C	AHB RX Data Buffer register (ARDB7)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
20	AHB RX Data Buffer register (ARDB8)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
24	AHB RX Data Buffer register (ARDB9)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
28	AHB RX Data Buffer register (ARDB10)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
2C	AHB RX Data Buffer register (ARDB11)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
30	AHB RX Data Buffer register (ARDB12)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
34	AHB RX Data Buffer register (ARDB13)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
38	AHB RX Data Buffer register (ARDB14)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
3C	AHB RX Data Buffer register (ARDB15)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
40	AHB RX Data Buffer register (ARDB16)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
44	AHB RX Data Buffer register (ARDB17)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
48	AHB RX Data Buffer register (ARDB18)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
4C	AHB RX Data Buffer register (ARDB19)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
50	AHB RX Data Buffer register (ARDB20)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
54	AHB RX Data Buffer register (ARDB21)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
58	AHB RX Data Buffer register (ARDB22)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
5C	AHB RX Data Buffer register (ARDB23)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
60	AHB RX Data Buffer register (ARDB24)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
64	AHB RX Data Buffer register (ARDB25)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>
68	AHB RX Data Buffer register (ARDB26)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/ 1707</a>

*Table continues on the next page...*

**memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
6C	AHB RX Data Buffer register (ARDB27)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/1707</a>
70	AHB RX Data Buffer register (ARDB28)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/1707</a>
74	AHB RX Data Buffer register (ARDB29)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/1707</a>
78	AHB RX Data Buffer register (ARDB30)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/1707</a>
7C	AHB RX Data Buffer register (ARDB31)	32	R/W	0000_0000h	<a href="#">30.4.5.5.1/1707</a>

**30.4.5.5.1 AHB RX Data Buffer register (ARDBn)**

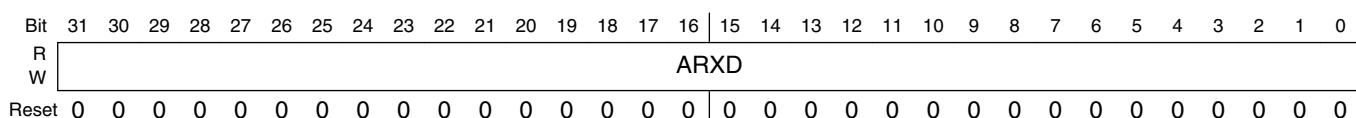
The AHB RX Data Buffer register 0 to 31 can be used to read the buffer content of the RX Buffer from successive addresses. QSPI\_ARDB0 corresponds to the RX Buffer register entry corresponding to the current value of the read pointer with increasing order.

The increment of the read pointer depends from the access scheme (DMA or flag-driven). Refer to "Data Transfer from the QuadSPI Module Internal Buffers" section in [Flash Read](#) section, RX Buffer, data read via register interface and AHB read, for the description of successive accesses to the RX Buffer content. Refer also to [Byte Ordering of Serial Flash Read Data](#) for the byte ordering scheme.

Valid address range accessible in the QSPI\_ARDBn range depends from the number of RX Buffer entries implemented and from the number of valid buffer entries available in the RX Buffer.

- Example 1, RX Buffer filled completely with 32 words: In this case the address range for valid read access extends from QSPI\_ARDB0 to QSPI\_ARDB31.
- Example 2, RX Buffer filled with 5 valid words, RX Buffer fill level QSPI\_RBSR[RDBFL] is 5. In this case an access to QSPI\_ARDB4 provides the last valid entry.

Address: 0h base + 0h offset + (4d × i), where i=0d to 31d

**ARDBn field descriptions**

Field	Description
ARXD	ARDB provided RX Buffer Data.

**ARDBn field descriptions (continued)**

Field	Description
	Byte order (endianness) is identical to the RX Buffer Data Registers.

## 30.5 Interrupt Signals

The interrupt request lines of the QuadSPI module are mapped to the internal flags according to the following table.

**Table 30-20. Assignment of Interrupt Request Lines**

IRQ/DMA line	QSPI_FR Flag	Interrupt Description
ipi_int_tfff	TBFF	TX Buffer Fill
ipi_int_tcf	TFF	Peripheral Command Transaction Finished
ipi_int_rfdf	RBDF	RX Buffer Drain
ipi_int_overrun		Buffer Overflow/Underrun Error Logical OR from:
	RBOF	RX Buffer Overrun
	TBUF	TX Buffer Underrun
	ABOF	AHB Buffer Overflow
ipi_int_cerr		Serial Flash Command Error Logical OR from:
	IPAEF	Peripheral access while AHB busy Error
	IPIEF	Peripheral Command could not be triggered Error
	IPGEF	Peripheral access while AHB Grant Error
	IUEF	Peripheral Command Usage Error
ipi_int_ored	TBFF, TFF, ILLINE, RBDF, RBOF, TBUF, ABSEF, ABOF, IPAEF, IPIEF, IPGEF, IUEF	Logical OR from all the QSPI_FR flags mentioned

## 30.6 Functional Description

This section provides the functional information of the QuadSPI module.

### 30.6.1 Serial Flash Access Schemes

The following access schemes are possible, depending on the serial flash devices attached to the QuadSPI module.

**Table 30-21. Access Schemes for Serial Flash Data Access**

Access Scheme	One Flash Device on Port A	One Flash Device on Port B	Two identical Flash Devices connected on Port A and Port B
<b>Individual Flash Mode: Access to Flash A</b>	Yes	N/a	Yes
<b>Individual Flash Mode: Access to Flash B</b>	N/a	Yes	Yes
<b>Parallel Flash Mode: Read from Flash A and Flash B</b>	N/a	N/a	Yes

**Note**

If two flash devices are accessed in Parallel Flash Mode, they are accessed with identical control signals. Special alignment on per-flash basis is **not** possible. It is within the responsibility of the application to ensure that the identical signals are applicable to both flash devices.

In Parallel Flash Mode, both external serial flash devices appear logically as one single memory doubled in size with respect to one individual flash device.

If two different flash devices are attached, they can be operated only in Individual Flash Mode.

In the Parallel Flash Mode, only data read commands are supported. Any other IP Command will result in an error condition signaled by the assertion of the QSPI\_FR[IUEF] flag and any other AHB Command will result in the assertion of the QSPI\_FR[ABSEF] flag.

In the Individual Flash Mode, all supported commands are available.

Unless explicitly noted, all the following descriptions relate to the Individual Flash Mode.

### 30.6.2 Modes of Operation

Refer to [QuadSPI Modes of Operation](#) for an overview over the possible operational modes of the QuadSPI block.

- Normal Mode can be used for write or read accesses to an external serial flash device.

- Serial Flash Write: Data can be programmed into the flash via the IP interface only. Refer to [Flash Programming](#) for further details.
- Serial Flash Read: Read the contents of the serial flash device. Two separate read channels are available via RX Buffer and AHB Buffer, see [Flash Read](#).
- Stop Mode: The mode is used for power management. When a request is made to enter Stop Mode, the QuadSPI block acknowledges the request and completes the SFM Command in progress, then the system clocks to the QuadSPI block may be shut off
- Module Disable Mode: The mode is used for power management. The clock to the non-memory mapped logic in the QuadSPI can be stopped while in Module Disable Mode. The module enters the mode by setting QSPI\_MCR[MDIS] or when a request is asserted by an external controller while QSPI\_MCR[DOZE] is set.

### 30.6.3 Normal Mode

This mode is used to allow communication with an external serial flash device. Compared to the standard SPI protocol, this communication method uses up to 4 bidirectional data lines operating at high data rates. The communication to the external serial flash device consists of an instruction code and optional address, mode, dummy and data transfers. The flexible programmable core engine described below is immune to a wide variety of command/protocol differences in the serial flash devices provided by various flash vendors.

#### 30.6.3.1 Programmable Sequence Engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands is given in the following table.

**Table 30-22. Instruction set**

Instruction	Instruction encoding	Pins	Operand	Action on Serial Flash(es)
CMD	6'd1	N=2'd{0,1,2} 2'd0 - One pad 2'd1 - Two pads	8 bit command value	Provide the serial flash with operand on the number of pads specified

*Table continues on the next page...*

**Table 30-22. Instruction set (continued)**

Instruction	Instruction encoding	Pins	Operand	Action on Serial Flash(es)
ADDR	6'd2	2'd2 - Four pads	Number of address bits to be sent (for example, 8'd24 => 24 address bits required)	Provide the serial flash with address cycles according to the operand on the number of pads specified. The actual address to be provided will be derived from the incoming address in case of AHB initiated transactions and the value of SFAR in case of IPS initiated transactions .
DUMMY	6'd3		Number of dummy clock cycles (should be <= 64 cycles)	Provide the serial flash with dummy cycles as per the operand. The PAD information defines the number of pads in input mode. (for example, one pad implies that pad 1 is not driven, rest all are driven)
MODE	6'd4		8 bit mode value	Provide the serial flash with 8 bit operand on the number of pads specified
MODE2	6'd5	N=2'd{0,1}	2 bit mode value	Provide the serial flash with 2 bit operand on the number of pads <sup>1</sup> specified
MODE4	6'd6	N=2'd{0,1,2}	4 bit mode value	Provide the serial flash with 4 bit operand on the number of pads <sup>2</sup> specified
READ	6'd7	N=2'd{0,1,2} 2'd0 - One pad 2'd1 - Two pads 2'd2 - Four pads	Read data size in bytes (for AHB transactions, the user's application should ensure that data size is a multiple of 8 bytes)	Read data from flash on the number of pads specified. The data size may be overwritten by writing to the ADATSZ field of the QSPI_BUFXCR registers for AHB initiated transactions and IDATSZ field of IP Configuration Register ( <a href="#">QuadSPI_IPCR</a> ) for IP initiated transactions.
WRITE	6'd8		Write data size in bytes	Write data on number of pads sepcified. The data size may be overwritten by writing to the IDATSZ field of IP Configuration Register ( <a href="#">QuadSPI_IPCR</a> ) register
JMP_ON_CS	6'd9	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion.
STOP	8'd0	NA	NA	Stop execution; deassert CS

- For a one pad instruction, MODE2 will take 2 serial flash clock cycles on the flash interface.
- For a one pad instruction, MODE4 will take 4 serial flash clock cycles on the flash interface. For a 4 pad instruction, MODE4 will take 1 serial flash clock cycle on the flash interface.

A sequence of such instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence pre-programmed in the LUT is referred to by its index.

The programmable sequence engine allows the user to configure the QuadSPI module according to the serial flash connected on board. The flexible structure is easily adaptable to new command/protocol changes from different vendors.

### 30.6.3.2 Flexible AHB buffers

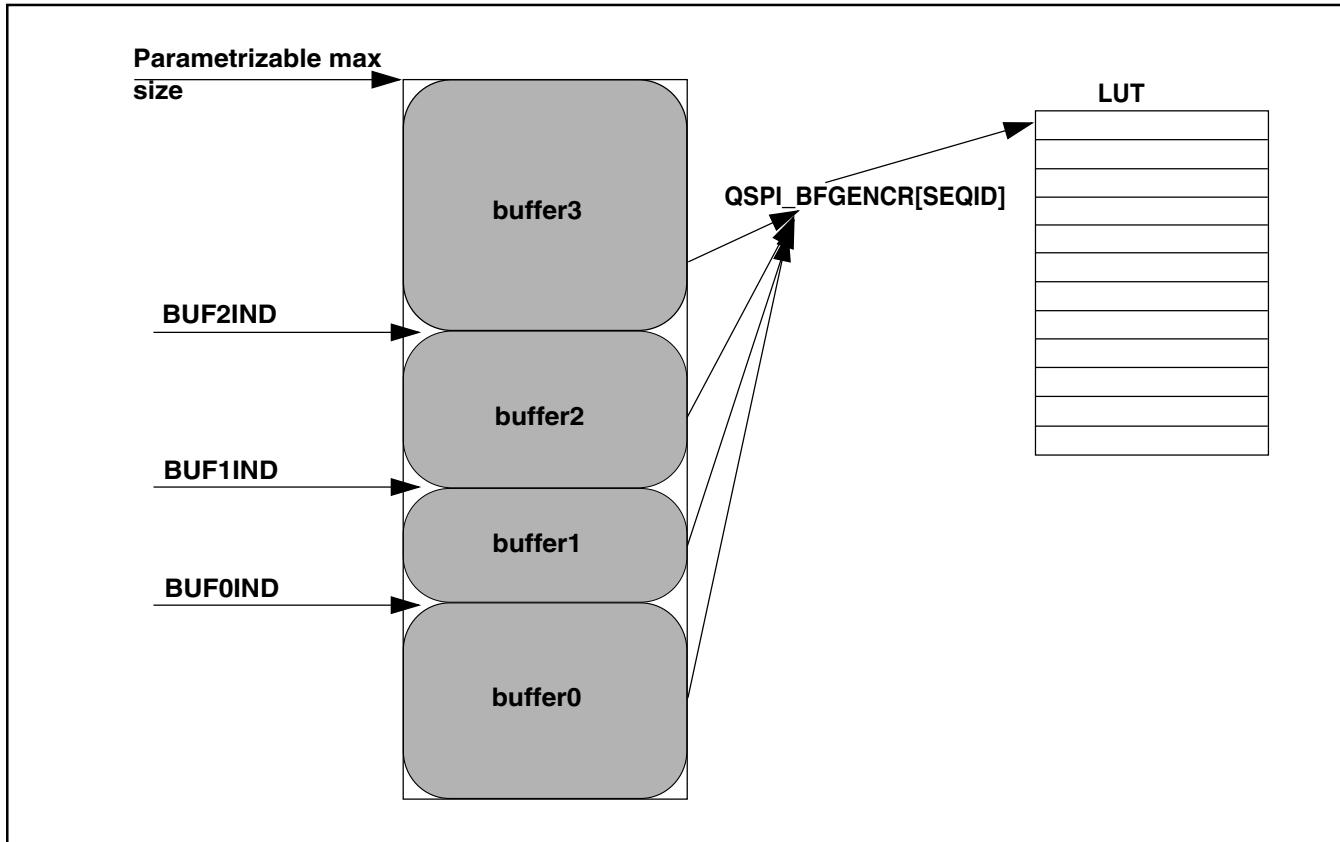
In order to reduce the latency of the reads for AHB masters, the data read from the serial flash is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 Bytes and maximum size being the size of the complete buffer instantiated. The size of buffer 0 is defined as being from 0 to QSPI\_BUF0IND. The Size of buffer 1 is from QSPI\_BUF0IND to QSPI\_BUF1IND, buffer2 is from QSPI\_BUF1IND to QSPI\_BUF2IND and buffer 3 is from QSPI\_BUF2IND to the size of the complete buffer, which is given in the chip-specific QuadSPI information.

Each flexible AHB buffer is associated with the following

1. An AHB master. Optionally, buffer3 may be configured as an "all master" buffer by setting the QSPI\_BUF3CR[ALLMST] bit. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
2. A datasize field representing the amount of data to be fetched from the flash on every "missed" access.

The master port number of every incoming request is checked and the data is returned/fetched into the corresponding associated buffer. Every "missed" access to the buffer causes the controller to clear the buffer and fetch QSPI\_BUFXCR[ADATSZ] amount of data from the serial flash. As such, there is no benefit in configuring a buffer size of greater than ADATSZ, as the locations greater than ADATSZ will never be used. For any AHB access, the sequence pointed to by the QSPI\_BFGENCR[SEQID] field is used for the flash transaction initiated. The data is returned to the master as soon as the requested amount is read from the serial flash. The controller however, continues to prefetch the rest of the data in anticipation of a next consecutive request. [Figure 30-3](#) shows the flexible AHB buffers.

The QSPI\_BFGENCR[SEQID] field points to an index of the LUT. Refer to [Look-up Table](#) for details.

**Figure 30-3. Flexible AHB Buffers**

Buffer0 may optionally be configured to be associated with a high priority master by setting the QSPU\_BUF0CR[HP\_EN] bit. An access by a high priority master suspends any ongoing prefetch to any of the other buffers. The ongoing prefetch is suspended and the high priority master is serviced first. Once the high priority masters access completes, the suspended transaction is resumed (before any other AHB access is entertained). The status of the suspended buffer can be read from [Sequence Suspend Status Register \(QuadSPI\\_SPNDST\)](#).

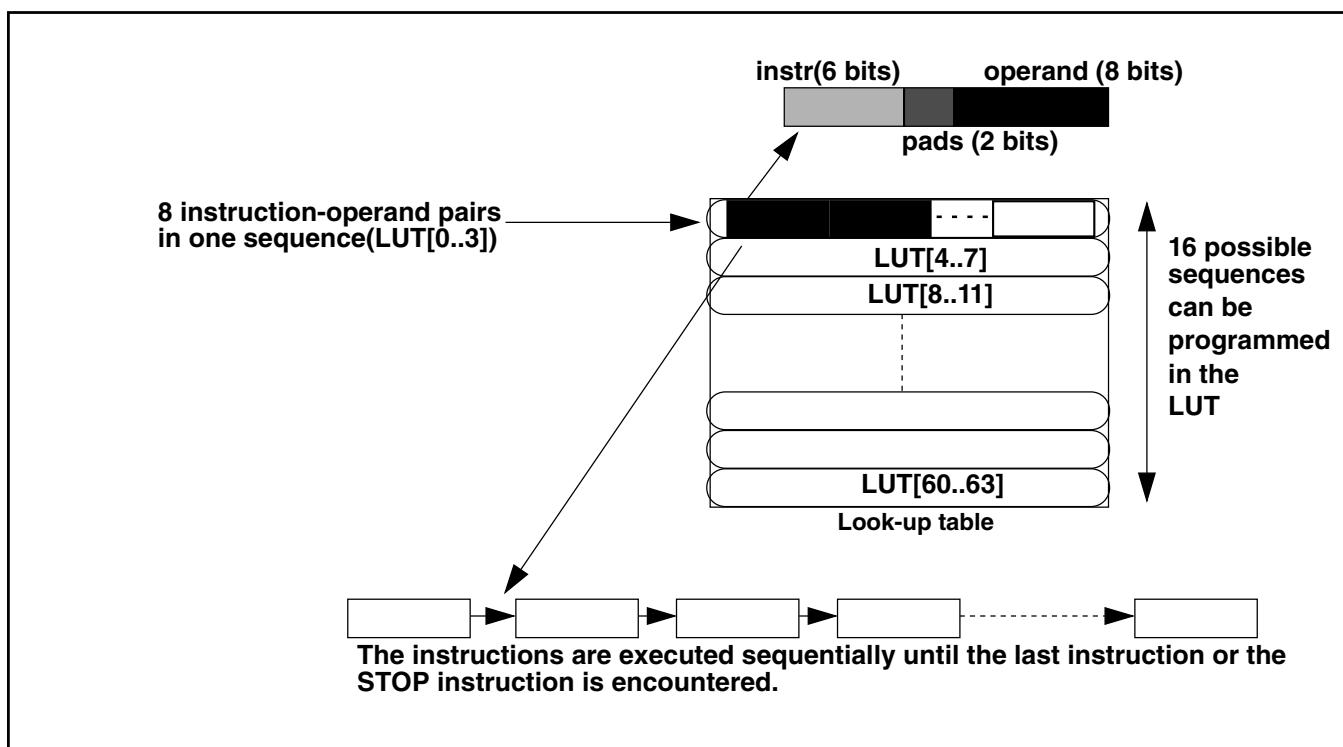
### 30.6.3.3 Suspend-Abort Mechanism

Any low priority AHB access can be suspended by a high priority AHB master request. The ongoing transaction is suspended at 64 bit boundary. The suspended transaction is restarted after the high priority master is served and the high priority transaction including data prefetch is completed. While a transaction is in suspended state, it may be aborted if a transaction by the same suspended master is made to a location which is different from the location of the suspended transaction.

Any ongoing transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

### 30.6.3.4 Look-up Table

The Look-up-table or LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs which when executed sequentially generates a valid serial flash transaction. Each sequence can have a maximum of 8 instruction-operand pairs. The LUT can hold a maximum of 16 sequences. The figure below shows the basic structure of the sequence in the LUT.



**Figure 30-4. LUT and sequence structure**

At reset, the index 0 of the look-up-table (LUT[0..3]) is programmed with a basic read sequence as given in [Table 30-23](#). After reset the complete LUT may be reprogrammed according to the device connected on board. In order to protect its contents during a code runover the LUT may be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is **0x5AF05AF0**. The process for locking and un-locking the LUT is as follows:

#### Locking the LUT

1. Write the key (**0x5AF05AF0**) in to the LUT Key Register (QuadSPI\_LUTKEY).

2. Write 0b01 to the [LUT Lock Configuration Register \(QuadSPI\\_LCKCR\)](#). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write into this register locks the LUT.

## Unlocking the LUT

1. Write the key (**0x5AF05AF0**) into the [LUT Key Register \(QuadSPI\\_LUTKEY\)](#)
2. Write 0b10 to the [LUT Lock Configuration Register \(QuadSPI\\_LCKCR\)](#). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write into this register unlocks the LUT.

The lock status of the LUT can be read from QSPI\_LCKCR[UNLOCK] and QSPI\_LCKCR[LOCK] bit.

Some example sequences are defined in [Example Sequences](#). The reset sequence at LUT index 0 is given in the following table.

**Table 30-23. Reset sequence**

Instruction	Pad	Operand	Comment
CMD	0x00	0x03	Read Data byte command on one pad
ADDR	0x00	0x18	24 Addr bits to be sent on one pad
READ	0x00	0x08	Read 64 bits
JMP_ON_CS	0x00	0x00	Jump to instruction 0 (CMD)

### 30.6.3.5 Issuing SFM Commands

Each access to the external device follows the same sequence:

1. The user must pre-populate the LUT with the serial flash command sequences that are required for the flash device being used.
2. The QuadSPI module starts executing the instructions in the sequence one by one. The transaction starts and the status bit QSPI\_SR[BUSY] is set.
3. Communication with the external serial flash device is started and the transaction is executed.

4. When the transaction is finished (all transmit- and receive operations with the external serial flash device are finished) the status bit QSPI\_SR[BUSY] is reset. In case of an IP Command the QSPI\_FR[TFF] flag is asserted.

Further details are given in below in [Flash Programming](#) and [Flash Read](#).

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

- **Using IP commands**

For IP Commands the required components need to be written into the following registers:

- Write the serial flash address to be used by the instruction into QSPI\_SFAR, refer to [Serial Flash Address Register \(QSPI\\_SFAR\)](#). For IP Commands not related to specific addresses, the base address of the related flash need to be programmed. For example, for an instruction which does not require an address (i.e. write enable instruction) the SFAR should be programmed with the base address of the memory the command is to be sent to.
- Write the sequence ID and data size details in the [IP Configuration Register \(QSPI\\_IPCR\)](#).
- Note that the write into the QSPI\_IPCR[SEQID] field must be the last step of the sequence. It is possible to combine all fields of the QSPI\_IPCR into one single write. Refer to [IP Configuration Register \(QSPI\\_IPCR\)](#) for details.

Note that there are some conditions where no IP Command is executed after writing the QSPI\_IPCR[SEQID] field and the write operation itself is ignored. They are described in [Command Arbitration](#).

- **Using AHB commands**

Any AHB memory mapped access is routed to one of the buffers depending on the master port number of the request. If the access is a "miss", a new serial flash transaction is started. The transaction is based on the sequence pointed to by the BFGENCR[SEQID] field as described in [Flexible AHB buffers](#).

An AHB access is termed memory mapped when the access is to the memory mapped serial flashes, as described in [Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A](#) and [Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B](#).

Again the possible error conditions are described in [Command Arbitration](#).

### 30.6.3.6 Flash Programming

In all cases the memory sector to be written needs to be erased first. The programming sequence itself is then initiated in the following way:

1. Check that the TX Buffer is empty. If the QSPI\_SR[TXNE]bit is set then the TX Buffer must be cleared by writing 1 into the QSPI\_MCR[CLR\_TXF] bit.
2. Program the address related to the command in the QSPI\_SFAR register.
3. Provide initial data for the program command into the circular buffer via register TX Buffer Data Register (QSPI\_TBDR) . At least one word of data must be written into the TX Buffer up to a maximum of 16.
4. Program the QSPI\_IPCR register to trigger the command. The QSPI\_IPCR[SEQID] should point to an index of the LUT which has the flash program sequence pre-programmed. The IDATSZ field should be set to denote the size of the write.
5. Depending on the amount of data required, step 3 must be repeated until all the required data have been written into the QSPI\_TBDR register. The QSPI\_SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, the QSPI\_TBSR[TRCTR] field can be read to check how many words have been written actually into the TX Buffer.

Upon writing the QSPI\_IPCR[SEQID] field (refer to step 4) the QuadSPI module will start to execute the programmed sequence. It is the responsibility of the software to ensure that a correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX Buffer. It consists of **16** entries of 32-bits and is organized as a circular FIFO, whose read pointer is incremented after each fetch. When all data are transmitted, the QuadSPI module will return from 'busy' to 'idle'. However, this is not true for the external device since the internal programming is still ongoing. It is up to the user to monitor the relevant status information available from the serial flash device and to ensure that the programming is finished properly.

### 30.6.3.7 Flash Read

Host access to the data stored in the external serial flash device is done in two steps. First, the data must be read into the internal buffers and in the second step these internal buffers can be read by the host.

#### 1. Reading Serial Flash Data into the QuadSPI Module Internal Buffers

A read access to the external serial flash device can be triggered in two different ways:

- **IP Command Read:** For reading flash data into the RX Buffer the user must provide the correct sequence ID in the QuadSPI\_IPCR[SEQID] register. The sequence ID points to a sequence in the LUT. It is the responsibility of the software to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash device connected on board. The user should program the Serial Flash Address Register (QSPI\_SFAR) and the IP Configuration Register (QSPI\_IPCR) registers. All available read commands supported by the external serial flash are possible.

Optionally it is possible to clear the RX Buffer pointer prior to triggering the IP Command by writing a 1 into the QuadSPI\_MCR[CLR\_RXF] field.

From these inputs, the complete transaction is built when the QSPI\_IPCR[SEQID] field is written. The transaction related to the read access starts and the requested number of bytes is fetched from the external serial flash device into the RX Buffer. Since the read access is triggered by an IP command, the IP\_ACC status bit and the BUSY bit are both set (both are located in the Status Register (QSPI\_SR) ). A count of the number of entries currently in the Rx Buffer can be obtained from QSPI\_RBSR[RDBFL].

The communication with the external serial flash is stopped when the specified number of bytes has been read (successful completion of the transaction).

- **AHB Command Read:** For reading flash data into the AHB Buffer the user must set up a read access by a master to the address range in the system memory map which the external serial flash devices are mapped to. The user should also program the buffer registers corresponding to the AHB master initiating the request, this depends on the configuration of the QSPI\_RBCT[RXBRD]. The user should provide the correct sequence ID into the buffer generic configuration register (QSPI\_BFGENCR). It is the responsibility of the software to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash device connected on board. Flash device selection and access mode are determined by the address accessed in the AHB address space associated to the QuadSPI module (refer to [Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A](#), [Memory Mapped Serial Flash Data - Individual Flash Mode on Flash B](#) and [Parallel Flash Mode](#).)

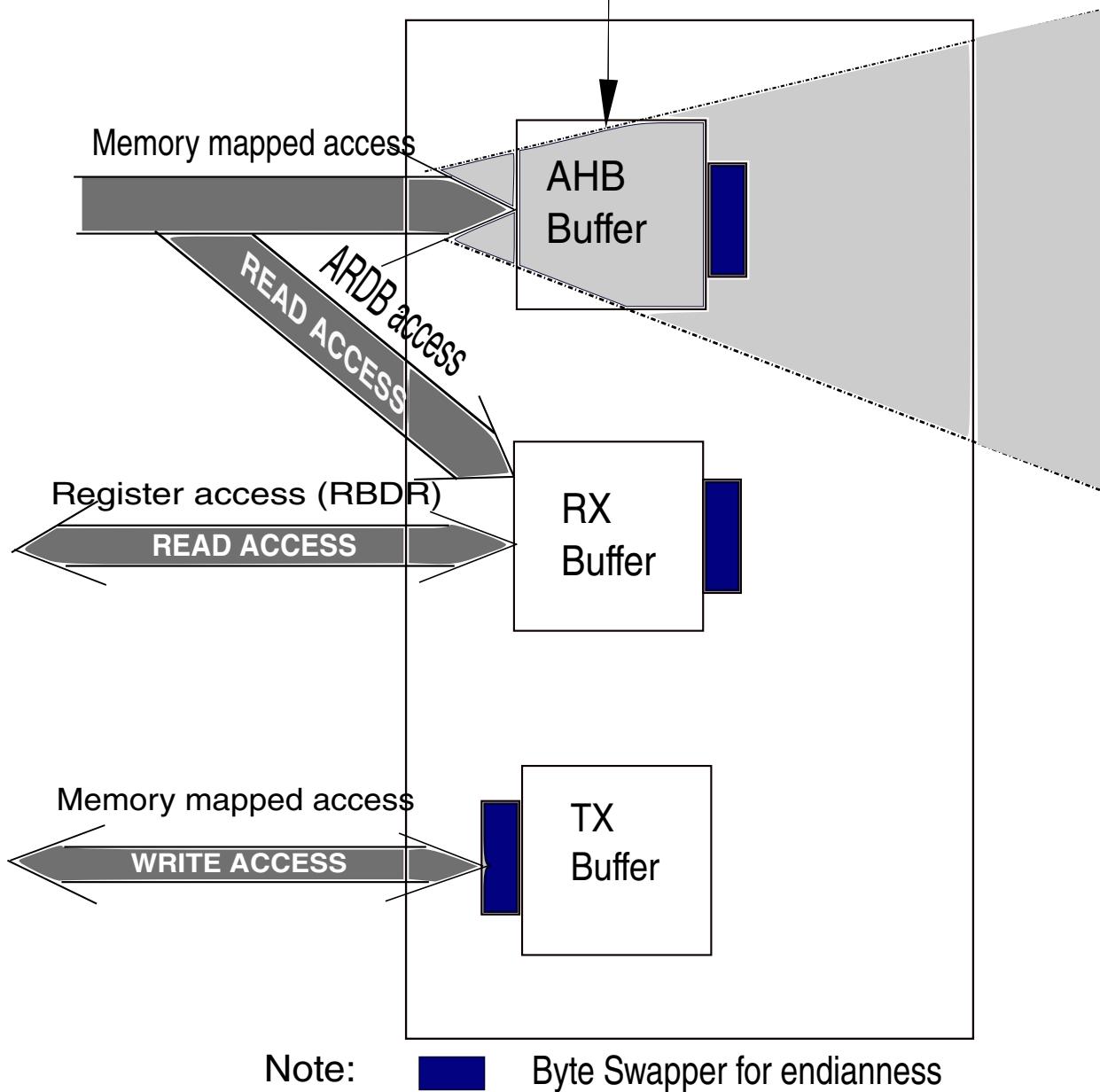
On each AHB read access to the memory mapped area the valid data in the AHB Buffer is checked against the address requested in the actual read. When the AHB read request can't be served from the content of the AHB Buffer, the buffer is flushed and the sequence pointed to by the sequence ID is executed by the

controller. The requested number of buffer entries defined in the QSPI\_BUFXCR[ADATSZ] field is then fetched from the external serial flash device into the internal AHB Buffer. Since the read access is triggered via the AHB bus, the QSPI\_SR[AHB\_ACC] status bit is set driving in turn the QSPI\_SR[BUSY] bit until the transaction is finished. The communication with the external serial flash is stopped when the specified number of entries has been filled.

## 2. Data Transfer from the QuadSPI Module Internal Buffers

The data read out from the external serial flash device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data has been loaded to. Refer to [Block Diagram](#) for details about the two available buffers, the RX Buffer and the AHB Buffer, in the QuadSPI module:

This Buffer is transparent to the user and is non-memory mapped



**Figure 30-5. QuadSPI memory map**

- The RX Buffer is implemented as FIFO of depth 32 entries of 4 bytes. Its content is accessible in two different address areas both referring to the identical data and the same physical memory.

In the IPS address space in the area associated to QSPI\_RBDR0 to QSPI\_RBDR31

In the AHB address space in the area associated to QSPI\_ARDB0 to QSPI\_ARDB31. Two successive entries are accessed with one single 64 bit AHB read operation.

RX Buffer operation can be summarized as follows: The QSPI\_RBCT[WMRK] field determines at which fill level the RXWE bit is asserted and how many entries are removed from the RX Buffer on each Buffer POP operation. So the QSPI\_SR[RXWE] bit indicates that the configured number of data entries is available in the RX Buffer and the QSPI\_RBSR[RDBFL] field indicates how many valid entries are available in total. Note that the first entry (QSPI\_RBDR0 or QSPI\_ARDB0) always corresponds to the first valid entry in the RX Buffer. The software needs to manage the number of valid data bytes itself.

Further details can be found in [RX Buffer Data Register \(QuadSPI\\_RBDRn\)](#) and in [AHB RX Data Buffer \(QSPI\\_ARDB0 to QSPI\\_ARDB31\)](#).

- **Flag-based Data Read of the RX Buffer** is done by polling the QSPI\_SR[RXWE] bit. When it is asserted the valid entries can be read either via the IPS address space (QSPI\_RBDRn) or the AHB address space (QSPI\_ARDBn). A Buffer POP operation must be triggered by the application by writing a 1 into the QSPI\_FR[RBDF] bit - this automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer will discard 16 bytes of data.
- **DMA controlled Data Read of the RX Buffer** is done by using the DMA module. The application must ensure that the DMA controller of the related device is programmed appropriately like it is described in [DMA Usage](#).

DMA controlled read out is triggered fully automatically by the assertion of the QSPI\_SR[RXWE] bit. The related Buffer POP operation is also handled completely inside the QuadSPI module. Like in the case above, accessing the RX Buffer content either on QSPI\_RBDRn or QSPI\_ARDBn related addresses is equivalent.

- **AHB Buffer data read via memory mapped access:** This kind of access is done by reading one of the addresses assigned to the external serial flash device(s) within the range given in [Table 30-16](#) table *under the condition that the data requested are already present in the AHB Buffer or it is currently being read from the serial flash device by the instruction in progress*. If this is not the case a memory mapped AHB command read is triggered as described above. If the requested data is already available in the AHB Buffer they are provided directly to the host.

When AHB access are made to the flash memory mapped address, the data will be fetched and returned to the AHB interface. Till the data is being fetched the AHB interface would be stalled. As soon as the data from the requested address has been read by the QuadSPI module the AHB read access is served. So it is possible to run sequential reads from the AHB buffer at arbitrary speed without

the need to monitor any information about the availability of the data. Nevertheless this access scheme stalls the AHB bus for the time required to read the data from the serial flash device. A better way is (when it is known the access is sequential) to have a prefetch enabled (by programming the ADATSZ field) such that before the next sequential AHB access come, the data is already fetched into the buffer.

As long as the host restricts its accesses to the data already in the buffer and the data currently fetched from the serial flash, it is possible to run the host read from the AHB Buffer in parallel to the serial flash read into the AHB Buffer.

### 30.6.3.8 Byte Ordering of Serial Flash Read Data

In this paragraph the byte ordering of the serial flash data is given. The basic scheme is that the **first** byte read out of the serial flash device - which is addressed by the QSPI\_SFAR[SFADR] field - corresponds to bit position QSPI\_RBDR0[0:7 ] register for IP Command read. In contrast to that for AHB Command read the bytes are always positioned according to the byte ordering of the AHB bus.

- **Byte Ordering in Individual Flash Mode**

The following table gives the byte ordering scheme of how the byte oriented data space of the serial flash device is mapped into one single 32 bit entry of the RX Buffer or the AHB Buffer. The table is valid within the following context:

- Flash A or Flash B in Individual Flash Mode
- All AHB data read commands with access size of 32 bit

**Table 30-24. Byte Ordering in Individual Flash Mode**

Serial Flash Byte Numbering	0	1	2	3
Buffer Entry Bit Position [31:0] (32 Bit data width)	[7:0]	[15:8]	[23:16]	[31:24]

#### Note

For IP Commands the read size can be given in number of bytes. If this number is not a multiple of 4, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

For AHB Commands, reads, starting from an address not aligned to 32 bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- **Byte Ordering in Parallel Flash Mode**

In Parallel Flash Mode each byte is combined out of 2 half bytes which are read in parallel from the two serial flash devices. The following tables shows how the flash content is separated into the half bytes and how the half bytes are assembled to the content of the QSPI\_RBDR0 register.

**Table 30-25. Serial Flash Device Half Byte Ordering**

Serial Flash Device Byte #	Flash A		Flash B	
	Bit Position [7:4]	Bit Position [3:0]	Bit Position [7:4]	Bit Position [3:0]
0	fah0	fal0	fbh0	fbl0
1	fah1	fal1	fbh1	fbl1
2	fah2	fal2	fbh2	fbl2
3	fah3	fal3	fbh3	fbl3
4	fah4	fal4	fbh4	fbl4
5	fah5	fal5	fbh5	fbl5
6	fah6	fal6	fbh6	fbl6
7	fah7	fal7	fbh7	fbl7
8	fah8	fal8	fbh8	fbl8

The table entry naming reflects the half byte positioning in the serial flash devices:

- <fa>h0 means **Flash A**, <fb>h0 means **Flash B**.
- fa<h>0 means half byte in **high position**, fa<l>0 means half byte in **low position**.
- fah<0> means **physical byte address 0** in the serial flash device, fal<1> means **physical byte address 1** in the serial flash device.

**Table 30-26. Byte Ordering in Parallel Flash Mode - RX Buffer**

QSPI_SFAR[SFADR] set to 0x000_0000
------------------------------------

*Table continues on the next page...*

**Table 30-26. Byte Ordering in Parallel Flash Mode - RX Buffer  
(continued)**

<b>QSPI_RBDR0</b>	fah0	hbh 0	fal0	fbl0	fah1	hbh 1	fal1	fbl1
<b>QSPI_ARDB0</b>								
<b>QSPI_SFAR[SFADR] set to 0x000_0001</b>								
<b>QSPI_RBDR0</b>	fah1	hbh 1	fal1	fbl1	fah2	hbh 2	fal2	fbl2
<b>QSPI_ARDB0</b>								
<b>QSPI_RBDR1</b>	fah3	hbh 3	fal3	fbl3	fah4	hbh 4	fal4	fbl4
<b>QSPI_ARDB1</b>								

**Note**

For IP Commands the read size can be given in number of bytes. If this number is not a multiple of 4 the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

**Table 30-27. Byte Ordering in Parallel Flash Mode - AHB Buffer**

<b>AHB Address + 0x4 (32 Bit Access)</b>	fah0	hbh 0	fal0	fbl0	fah1	hbh 1	fal1	fbl1
<b>AHB Address 0x800_0004 (32 Bit Access)</b>	fah2	hbh 2	fal2	fbl2	fah3	hbh 3	fal3	fbl3

**Note**

For AHB Command read starting from an address not aligned to 32 bit boundaries or AHB access size smaller than 32 bit the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- **Buffer Entry Ordering for 64 Bit Read Access**

For read access via the AHB interface 64 bit access is possible. Each 64 bit access reads 2 32 bit entries simultaneously. The ordering of these 32 bit entries within the 64 bit word is given in the following table.

**Table 30-28. 64 Bit Read Access Buffer Entry Ordering**

AHB Read Data Bit Position [63:0]	[31:0]	[63:32]
Buffer Entry #	Even (0, 2, 4, ...)	Odd (1, 3, 5, ...)

### 30.6.3.9 Normal Mode Interrupt and DMA Requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are related to the [Flag Register \(QSPI\\_FR\)](#).

**Table 30-29. Interrupt and DMA Request Conditions**

Condition	Flag(QSPI_FR)	DMA
TX Buffer Fill	TBFF	-
TX Buffer Underrun	TBUF	-
Illegal Instruction Error	ILLINE	-
RX Buffer Drain	RBDF	X
RX Buffer Overflow	RBOF	-
AHB Buffer Overflow	ABOF	-
AH Sequence Error	ABSEF	-
IP Command Usage Error	IUEF	-
IP Command Trigger during AHB Access Error	IPADEF	-
IP Command Trigger could not be executed Error	IPIEF	-
IP Access during AHB Grant Error	IPGEF	-
IP Command related Transaction Finished	TFF	-

Each condition has a flag bit in the [Flag Register \(QSPI\\_FR\)](#) and a Request Enable bit in the [DMA Request Select and Enable Register \(QSPI\\_RSER\)](#). The RX Buffer Drain Flag (RBDF) has separate enable bits for generating IRQ and DMA requests. Note that not all flags have an individual IRQ line. Check the devices Interrupt Vector Table for more details.

- Transmit Buffer Fill Interrupt Request:

The Transmit Buffer Fill IRQ indicates that the TX Buffer can accept new data. It is asserted if the QSPI\_FR[TBFF] flag is asserted and if the corresponding enable bit (QSIP\_RSER[TBFIE]) is set. Refer to [TX Buffer Operation](#), for details about the assertion of the QSPI\_FR[TBFF] flag.

- Receive Buffer Drain Interrupt or DMA Request:

The Receive Buffer Drain IRQ derived from the QSPI\_FR[RBDF] flag indicates that the RX Buffer of the QuadSPI module has data available from the serial flash device to be read by the host. It remains set as long as the QSPI\_RBSR[RXWE] bit is set. The QSPI\_RSER[RBDIE] bit enables the related IRQ.

Aside from the IRQ it is possible to handle RX Buffer drain by DMA. If the QSPI\_RSER[RBDDE] bit is set, a DMA request will be triggered when the RX Buffer contains more than QSPI\_RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer Overflow/Underrun Interrupt Request:

The Buffer Overflow/Underrun IRQ is a combination of the following flags (all located in the QSPI\_FR register with the related enable bits in the QSPI\_RSER register):

- TBUF - TX Buffer Underrun, enabled by TBUIE
- RBOF - RX Buffer Overflow, enabled by RBOIE
- ABOF - AHB Buffer Overflow, enabled by ABOIE

The Transmit Buffer Underrun indicates that an underrun condition in the TX Buffer has occurred. It is generated when a write instruction is triggered whilst the Tx Buffer is empty and the QSPI\_RSER[TBUIE] bit is set.

The Receive Buffer Overflow indicates that an overflow condition in the RX Buffer has occurred. It is generated when the RX Buffer is full, an additional read transfer attempts to write into the RX Buffer and the QSPI\_RSER[RBOIE] bit is set.

The AHB Buffer Overflow indicates that an overflow condition in the AHB Buffer has occurred. It is generated when the AHB Buffer is full, an additional read transfer attempts to write into the AHB Buffer and the QSPI\_RSER[ABOIE] bit is set.

The data from the transfers that generated the individual overflow conditions is ignored.

- Serial Flash Command Error Interrupt Request

If the IPAEF, IPIEF, IPGEF or IUEF flags in the QSPI\_FR are set, and the related interrupt enable bits in the QSPI\_RSER are also set, then an interrupt is requested.

- Transaction Finished Interrupt Request

The IP Command Transaction Finished IRQ indicates the completion of the current IP Command. It is triggered by the QSPI\_FR[TFF] flag and is masked by the QSPI\_RSER[TFIE] bit.

### 30.6.3.10 TX Buffer Operation

The TX Buffer provides the data used for page programming. For proper operation it is required to provide at least one entry in the TX Buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX Buffer fast enough as long as the command is executed without a TX Buffer overflow or underrun.

The QuadSPI module sets the QSPI\_FR[TBFF] flag so long as the TX Buffer is not full and can accept more data.

When the QuadSPI module tries to pull data out of an empty TX Buffer the TX Buffer underrun is signaled by the QSPI\_FR[TBUF] flag. The current IP Command leading to the underrun condition is continued until the specified number of bytes has been sent to the serial flash device, in the underrun condition when QuadSPI module tries to pull out data of empty TX buffer, the data transferred is undefined i.e. once the underrun flag is set, it will return the garbage value until the required number of bytes are not sent. When this Sequence Command is finished, the QSPI\_FR[TBFF] flag is asserted indicating that the Tx Buffer is ready to be written again.

The TX Buffer overflow isn't signaled explicitly, but the TX Buffer fill level can be monitored by the QSPI\_TBSR[TRBFL] field.

Refer to [TX Buffer Status Register \(QuadSPI\\_TBSR\)](#) and [Flag Register \(QuadSPI\\_FR \)](#) for details about the TX Buffer related registers.

### 30.6.3.11 Address scheme

Earlier serial flash memories supported only 24-bit address space hence restricting the maximum memory size of the serial flash as 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to legacy 24-bit address mode.

- **Extended Address Mode**

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR command should be programmed with 8'd32 as the operand value. By default, the QuadSPI is in 24-bit legacy address mode. Each of the memory vendors have a different way of enabling this mode (Refer to the memory specification from memory vendors). For example, the command B7h sent to Macronix flash will enable it for 32-bit address mode.

- **Extended Address register**

In this mode, the upper 8-bit of the 32-bit address is provided by the Extended address register in the memory itself. The memory provides a specific register which is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB, consists of banks of 16 MB. The 8-bit written in the extended address register effectively enables a bank. For example in Spansion memory, when the extended address register is updated with a value of 0x01 with the help of the command 17h, it will open Bank1 of the memory. The consequent 24-bit address commands will lead to Bank1. The extended address register needs to be update with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

## 30.7 Initialization/Application Information

This section provides the initialization and application information of the QuadSPI module.

### 30.7.1 Power Up and Reset

Note that the serial flash devices connected to the QuadSPI module may require special voltage characteristics of their inputs during power up or reset. It is the responsibility of the application to ensure this.

## 30.7.2 Available Status/Flag Information

This paragraph gives an overview of the different status and flag information available and their interdependencies for different use cases. Related registers are QSPI\_SR and QSPI\_FR. Refer to the related descriptions how to set up the QuadSPI module appropriately.

### 30.7.2.1 IP Commands

Refer to [IP Configuration Register \(QuadSPI\\_IPCR\)](#) for additional details not explicitly covered in this paragraph.

- **IP Commands - Normal Operation**

Writing the QSPI\_IPCR[SEQID] field triggers the execution of a new IP Command. Given that this is a legal command the QSPI\_SR[IPACC] and the QSPI\_SR[BUSY] bits are asserted simultaneously, immediately after the execution is started.

When the instruction on the serial flash device has been finished these bits are de-asserted and the QSPI\_FR[TFF] flag is set.

- **IP Commands - Error Situations**

Refer to [Table 30-30](#) below.

### 30.7.2.2 AHB Commands

Refer to Section 1, Reading Serial Flash Data into the QuadSPI Module, in [Flash Read](#) for additional details not explicitly covered in this paragraph.

- **AHB Commands - Normal Operation**

Memory mapped read access to a serial flash address not contained in the AHB Buffer, triggers the execution of an AHB Command. Given that this is a legal command the QSPI\_SR[AHBACC] and the QSPI\_SR[BUSY] bits are asserted simultaneously immediately after the execution is started. When the instruction on the serial flash device has been finished these bits are de-asserted.

- **IP Commands - Error Situations**

Refer to [Table 30-30](#) below.

### 30.7.2.3 Overview of Error Flags

The following table gives an overview of the different error flags in the QSPI\_FR register and additional error-related details.

**Table 30-30. Overview of QSPI\_FR Error Flags**

Error Category	Error Flag in QSPI_FR	Command Execution on Serial Flash Device TFF Behavior (in case of IP commands only)	Description
<b>AHB Error Flag</b>	ABSEF	Flash transaction is aborted	AHB sequence contains <ul style="list-style-type: none"> <li>• WRITE instruction</li> </ul>
<b>AHB Error Flag</b>	ABOF	Flash transaction continues until it finishes	Set when the module tried to push data into the AHB buffer that exceeded the size of the AHB buffer. Only occurs due to wrong programming of the QSPI_BUFXCR[ADATSZ].
<b>Miscellaneous Error Flag</b>	ILLINE	Flash transaction aborted	Illegal instruction Error Flag – Set when an illegal instruction is encountered by the controller in any of the sequences.
<b>Command Arbitration Error</b>	IPIEF	TFF not asserted in conjunction with that command	IP Command Error - caused when IP access is currently in progress (IP_ACC set) and <ul style="list-style-type: none"> <li>• write attempt to QSPI_IPCR register.</li> <li>• write attempt to QSPI_SFAR register.</li> <li>• write attempt to QSPI_RBCT register.</li> </ul>
<b>Command Arbitration Error</b>	IPAEF		<ul style="list-style-type: none"> <li>• AHB Command already running, another IP Command could not be executed.</li> <li>• AHB Command already running, write attempt to QSPI_IPCR[SEQID] field.</li> </ul>
<b>Command Arbitration Error</b>	IPGEF		<ul style="list-style-type: none"> <li>• Exclusive access to the serial flash granted for AHB Commands, write attempt to QSPI_IPCR[SEQID] field.</li> </ul>
<b>IP Command Error</b>	IUEF	—	<ul style="list-style-type: none"> <li>• IP Command Usage Error</li> </ul>
<b>Buffer Related Error</b>	RBOF	TFF is asserted on completion	<ul style="list-style-type: none"> <li>• RX Buffer Overrun</li> </ul>
<b>Buffer Related Error</b>	TBUF		<ul style="list-style-type: none"> <li>• TX Buffer Underrun</li> </ul>

Note that only the buffer related errors are related to a transaction on the external serial flash. All the other errors do not trigger an actual transaction.

### 30.7.2.4 IP Bus and AHB Access Command Collisions

There are two flags related to this topic, the QSPI\_FR[IPAEF] and QSPI\_FR[IPIEF]. Refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of [Flash Read](#) section, for a description of the flags and [Command Arbitration](#), for details about possible command collisions.

### 30.7.3 Exclusive Access to Serial Flash for AHB Commands

It is possible that several masters need to access the serial flash device connected to the QuadSPI module separately, one master by triggering IP Commands and reading the RX Buffer (via RBDR $n$  register) and the other masters by triggering AHB Commands (via ARDB $n$  Registers). These two set of buffer (RBDR and ARDB Buffer) points to the same physical buffer. Refer to [Figure 30-5](#) To avoid command collisions resulting in excessive latencies the QuadSPI module implements a request-handshake mechanism between the master triggering AHB Commands and the QuadSPI module allowing this specific master to request exclusive access to the serial flash device for AHB Commands. If this exclusive access is granted the execution of IP Commands is blocked. This resolves command collisions and excessive times where the AHB interface may be blocked.

If this capability is used in the device there is additional status and flag information available related to this mechanism. The QSPI\_SR[AHBGNT] bit reflects the module-internal state that the exclusive access mentioned above is granted, any attempt to trigger an IP Command is rejected and results in the assertion of the QSPI\_FR[IPGEF] flag. Refer to the descriptions of the related bit and flag for details.

It is within the responsibility of the application to set up the master using this mechanism appropriately, if used incorrectly no IP Commands at all can be triggered.

Two different cases can be distinguished:

#### 30.7.3.1 RX Buffer Read via QSPI\_ARDB Registers

In this case all masters share the AHB bus for RX Buffer as well as for AHB Buffer read. In this case the access to the AHB interface by the master triggering AHB Commands must be deferred until any pending IP Command has been finished **and** the RX Buffer readout has been finished as well. The QSPI ARDB Buffers access the Rx buffer i.e the

data from the Rx Buffer is returned and no data from AHB Buffer is touched. This is the conservative use case, corresponding to the reset value 0 of the QSPI\_RBCT[RXB RD] bit.

In this case the QSPI\_SR[AHBGNT] bit is asserted not earlier than any running IP Command has been finished (QSPI\_SR[IP\_ACC] is 0), the RX Buffer has been read out completely (QSPI\_RBSR[RDBFL] equal to 0) or no DMA read is pending (QSPI\_SR[RXDMA] equal to 0 and Rx Buffer readout is via AHB(QSPI\_RBCT[RXB RD]) equal to 1.

### 30.7.3.2 RX Buffer Read via QSPI\_RBDR Registers

This is the preferred use case as an access to the AHB buffer (memory mapped flash) does not interfere with any IPS access to read the RBDR buffer. It is not possible that a pending AHB bus access triggered by an AHB Command stalls the AHB bus and blocks the RX Buffer readout since the RX Buffer is read via the IP bus based registers QSPI\_RBDR0 to QSPI\_RBDR31.

For this case it is recommended to program the QSPI\_RBCT[RXB RD] bit to 1. The QSPI\_SR[AHBGNT] bit is asserted immediately after any running IP Command has been finished (QSPI\_SR[IP\_ACC] is 0), the RX Buffer has been read out completely (QSPI\_RBSR[RDBFL] equal to 0) or no DMA read is pending (QSPI\_SR[RXDMA] equal to 0, allowing the master triggering AHB Commands to trigger AHB Commands as soon as possible without the need to wait for the RX Buffer readout to be finished.

### 30.7.4 Command Arbitration

In case of overlapping commands, the arbitration scheme is described in the following paragraphs under the assumption that the priority mechanism described in [Exclusive Access to Serial Flash for AHB Commands](#) is **not** used:

- During the execution of an IP Command, the running IP Command can't be terminated by issuing another IP Command or AHB Command. The QSPI\_FR[IPIEF] flag is asserted when the host tries to write into the QSPI\_IPCR register. When the host triggers an AHB Command (refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of [Flash Read](#) section, for details), this command is stalled until the currently running IP Command is finished.
- During the execution of an AHB Command, the running AHB Command can't be terminated by issuing an IP Command. The command is ignored and the QSPI\_FR[IPA EF] flag is asserted. Refer to [Flag Register \(QuadSPI\\_FR \)](#) for the description of these flags.

When another AHB Command is triggered the address of the memory mapped access is considered. If the requested address is currently read from the serial flash device, the running command is continued. If this is not the case the currently running command is terminated and another AHB Command related to the requested address is executed. Refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of [Flash Read](#) section, for further details.

In case of coinciding commands the IP Command is triggered and the AHB Command is stalled until the IP Command has been finished (QSPI\_SR[IP\_ACC] has been deasserted).

The IP Commands ignored in case of command collision will not result in the assertion of the QSPI\_FR[TFF] flag.

### 30.7.5 Flash Device Selection

Regardless of the SFM Command (IP or AHB) the access mode is selected by specifying the 32 bit address value for the following SFM Command.

For IP Commands the access mode is selected with the address programmed into the QSPI\_SFAR register. Refer to [Serial Flash Address Register \(QuadSPI\\_SFAR\)](#) for details.

For AHB Commands the access mode is determined by the memory mapped address which is accessed Refer to [AMBA Bus Register Memory Map](#) for details.

### 30.7.6 DMA Usage

For the complete description of the DMA module refer to the related DMA Controller chapter. In this paragraph only the details specific to the DMA usage related to the QuadSPI module are given.

#### 30.7.6.1 DMA Usage in Normal Mode

### 30.7.6.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash -> AHB bus / IP Bus -> DMA controller) involved in the read data process is essential for proper operation. Such analysis must take into account not only the data rate provided by the serial flash but also the data rate of the AHB bus and the performance of the DMA controller in reading data from the RX buffer.

Two figures must match for proper operation, that means that the data rate provided by the serial flash device must not exceed the average RX Buffer readout data rate. Otherwise, the longer this state persists, a RX Buffer overflow will result.

#### AHB Bus Side (data read):

The total number of bus cycles for each DMA Minor Loop completion is added from the following components:

- Overhead for each minor loop, given by DMA controller: Assume 10 cycles
- Overhead due to clock domain crossing: Assume 2 cycles
- Number of bus clock cycles required for 8 bytes (64 bit read size): Assume 2 cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of the QSPI\_RBCT[WMRK] field, therefore the overhead given above distributes among (QSPI\_RBCT[WMRK]+1)/2 read accesses of 64 bit each.

The following table gives some examples for typical use cases:

**Table 30-31. Access Duration Examples - Bus Clock Side**

QSPI_RBCT[WMRK]	Number of Bytes per DMA Loop <sup>1</sup>	Number of Bus Clock cycles for DMA Minor Loop	Time Duration of DMA Minor Loop for 120Mhz Bus clock Frequency
0	4	12+2 = 14	~117ns
1	8	12+2 = 14	~117ns
3	16	12+4 = 16	~133ns
7	32	12+8 = 20	~167ns
11	48	12+12 = 24	~200ns

1. DMA Loop means one Minor Loop Completion which is equivalent to one.

#### NOTE

The table figure represents ideal scenario, actual performance will depend on how the system is integrated.

#### Serial Flash Device Side (data read):

The number of serial flash cycles can be determined in the following way:

- Number of serial flash clock cycles required to read 4 bytes, corresponding to one RX Buffer entry (setup of command and address not considered): 4 cycles for Quad (SDR) mode instruction in parallel flash mode, 8 cycles for Quad Mode (SDR) instructions in Individual Flash Mode etc.
- Overhead due to clock domain crossing : 1 cycle.

The following table lists the number of clock cycles required to read the data from the serial flash corresponding to the different settings of the QSPI\_RBCT[WMRK] field:

**Table 30-32. Access Duration Examples - Serial Flash side**

QSPI_RBCT[WMRK] setting	Num Bytes per DMA Loop <sup>1</sup>	Num SCKFx for 60MHz SCKFx	Time duration of Flash data readout for 60MHz SCKFx (~16.6ns period)
		IFM <sup>2</sup> Quad	IFM Quad
0	4	9	~150ns
1	8	17	~282ns
3	16	33	~548ns
7	32	65	~1079ns
11	48	97	~1610ns

1. DMA Loop means one Minor loop completion which is equivalent to one Major Loop iteration.
2. Individual flash mode.

From the examples given in the two tables above, it can be seen that depending on the relationship between the Bus clock and Serial flash clock frequencies, there are settings possible where the serial flash provides the read data faster than the AHB bus can read out the RX buffer. In these cases, the RX buffer data keeps accumulating over time and will eventually overflow. To avoid RX Buffer overflow, the data transaction size should be small enough.

A complementary example would be when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be smaller than the time taken by the controller to push in the remaining entries in the buffer.

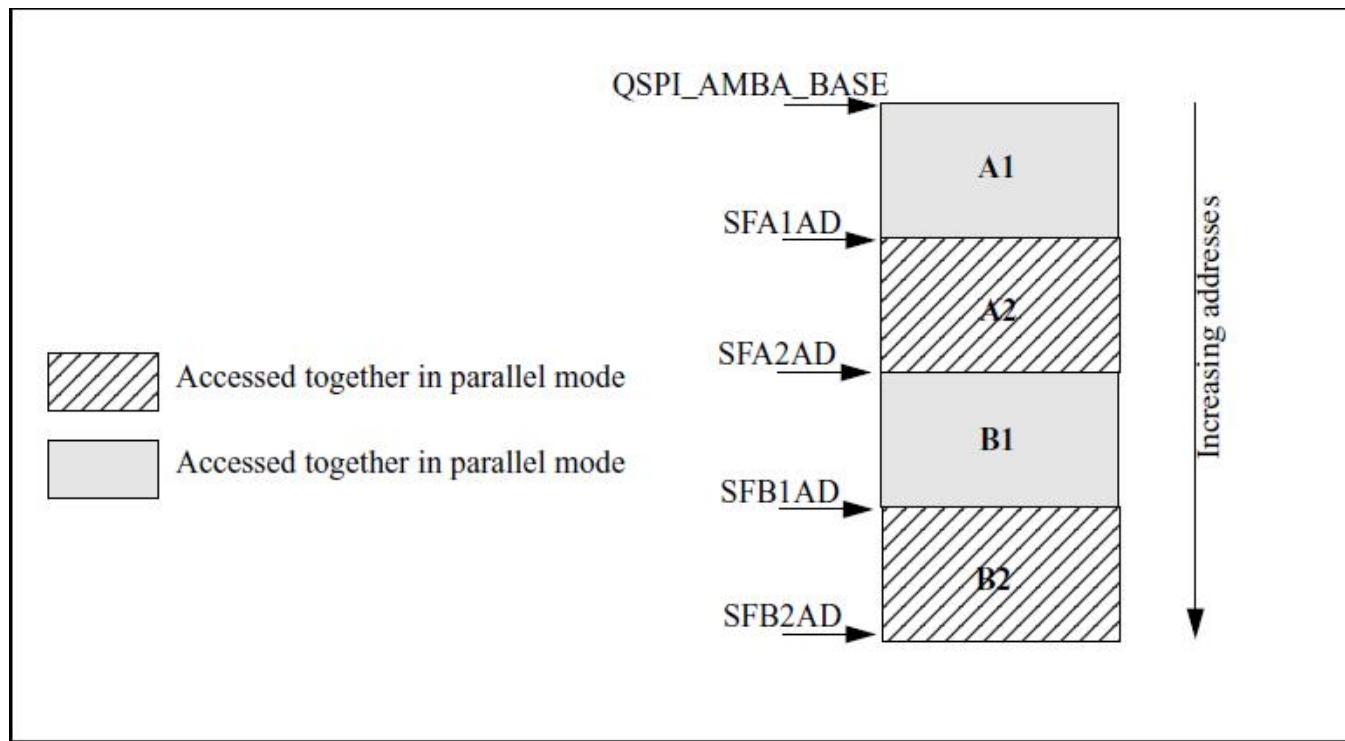
#### NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

### 30.7.7 Parallel mode

QuadSPI can access two flashes in parallel. This increases the throughput of the QuadSPI by two times. Only read operations are allowed in parallel mode. In case a write transaction is initiated in parallel mode, QSPI\_FR[IUEF] is set. When dual die flashes are

accessed in parallel mode, it is mandatory for flash A1 to be of the same size as B1 and A2 to be of the same size as B2. The following figure shows how QuadSPI maps the incoming addresses to the different flashes connected on board.



**Figure 30-6. Flash addressing**

An example programming for parallel mode access is given below (flash sizes are assumed to be 256MB):

- QSPI\_AMBA\_BASE - 0x10000000
- QSPI\_SFA1AD[TPADA1] - 0x20000000
- QSPI\_SFA2AD[TPADA2] - 0x30000000
- QSPI\_SFB1AD[TPADB1] - 0x40000000
- QSPI\_SFB2AD[TPADB2] - 0x50000000

In order to access the first location of A1/B1 pair, the incoming address should be 0x10000000. QSPI\_AMBA\_BASE is subtracted from this address and the result is divided by two. Therefore, address provided to flash A1 and B1

$$\text{Flash Address} = (\text{Memory mapped address} - \text{QSPI_AMBA_BASE})/2$$

For Memory Mapped address:

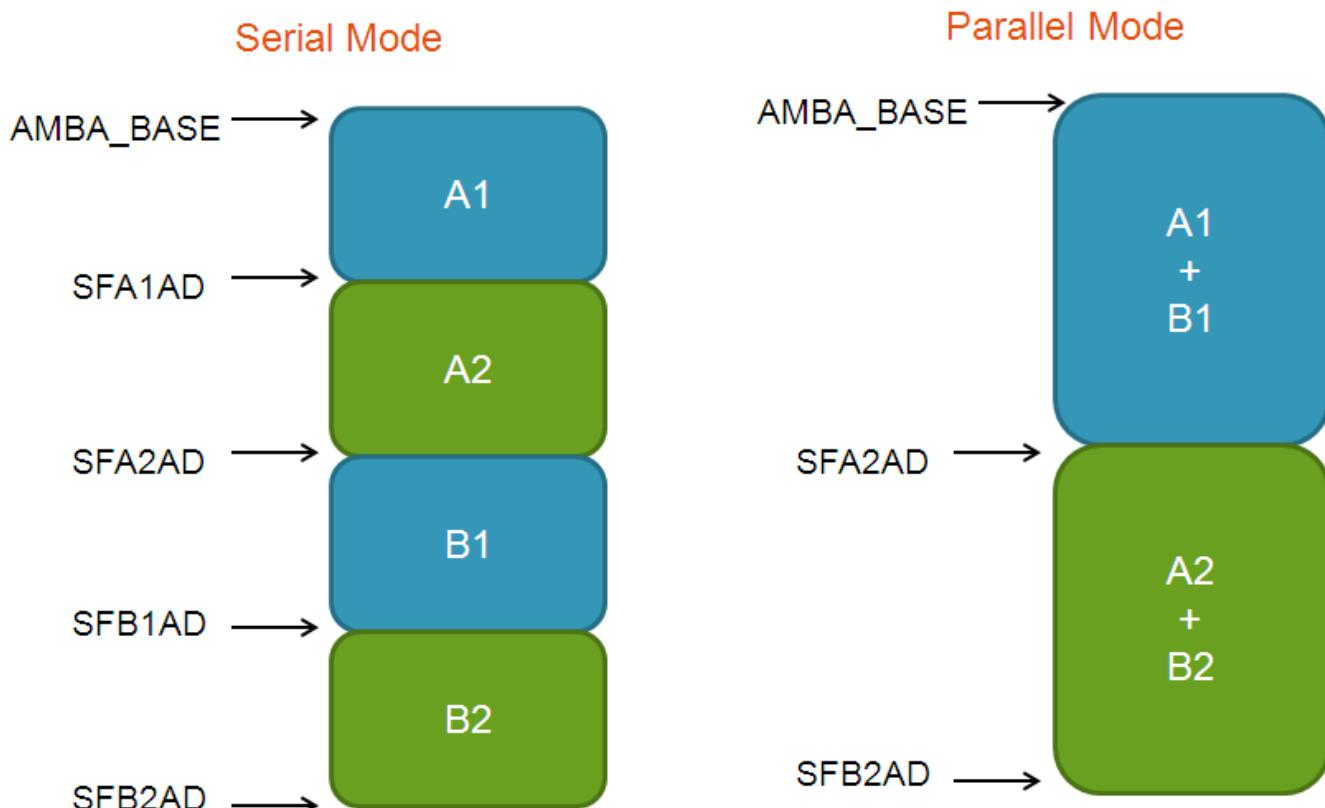
- 0x10000000, flash address: 0x0 (Or, the first address of flash A1 and B1)
- 0x10000004, flash address: 0x2
- 0x10000008, flash address: 0x4 etc.

Similarly, in order to access the first location of A2/B2 pair, the incoming address should be 0x30000000.

$$\text{Flash Address} = (\text{Memory mapped address} - \text{SFA2AD})/2$$

For Memory Mapped address:

- 0x30000000, flash address: 0x0 (Or, the first address of flash A2 and B2)
- 0x30000004, flash address: 0x2
- 0x30000008, flash address: 0x4 etc.



**Figure 30-7. Memory map - Serial and Parallel**

## 30.8 Byte Ordering - Endianness

QuadSPI provides support for swapping the flash read/write data based on the configuration of the QSPI\_MCR[END\_CFG]. By default the data is always returned in 64 bit BE format on the AHB bus and 32 bit BE format on the IPS interface when read via the RX buffer and written in 32 bit BE format when written via the TX buffer.

#### Byte Ordering - Endianness

The table(QSPI\_MCR[END\_CFG]) below shows the complete bit ordering. BE signifies Big Endian which means the high order bits of the associated data vectors are associated with low order address positions. LE signifies Little Endian which means the lower order bits of the associated data vectors are associated with low order address positions. Refer to figure [Figure 30-5](#)

**Table 30-33. QSPI\_MCR[END\_CFG]**

00	64 bit BE
01	32 bit LE
10	32 bit BE
11	64 bit LE

The tables below (Byte ordering configuration in AHB) and (Byte ordering configuration in IPS) show how this configuration is implemented in QSPI AHB and IPS interfaces respectively. B in the table signifies Byte and the index 1-8 refers to the byte position i.e. 1 refer to bits[7:0], 8 refer to bits[63:56] and so on.

**Table 30-34. Byte ordering configuration in AHB**

64 bit BE	B1	B2	B3	B4	B5	B6	B7	B8
64 bit LE	B8	B7	B6	B5	B4	B3	B2	B1
32 bit BE	B5	B6	B7	B8	B1	B2	B3	B4
32 bit LE	B4	B3	B2	B1	B8	B7	B6	B5

**Table 30-35. Byte ordering configuration in IPS**

32BE	B1	B2	B3	B4
32LE	B4	B3	B2	B1

The examples below show the byte ordering in 64 bit BE configuration for AHB Buffer and 32 bit BE for TX/RX Buffer:

### 30.8.1 Programming Flash Data

CPU write instructions to the QSPI\_TBDR register like

- Write QSPI\_TBDR -> 0x01\_02\_03\_04
- Write QSPI\_TBDR -> 0x05\_06\_07\_08

result in the following content of the TX Buffer:

**Table 30-36. Example of QuadSPI TX Buffer**

TX Buffer Entry	Content
0	32'h01_02_03_04
1	32'h05_06_07_08

Programming the TX Buffer into the external serial flash device results in the following byte order to be sent to the serial flash:

- 01...02...03...04...05...06...07...08

### 30.8.2 Reading Flash Data into the RX Buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

This results in the RX Buffer filled with:

**Table 30-37. Resulting RX Buffer Content**

RX Buffer Entry	Content
0	32'h01_02_03_04
1	32'h05_06_07_08

#### 30.8.2.1 Readout of the RX Buffer via QSPI\_RBDRn

The RX Buffer content appears at CPU read access via the Peripheral bus interface in the following order:

- Read QSPI\_RBDR0 <- 0x01\_02\_03\_04
- Read QSPI\_RBDR1 <- 0x05\_06\_07\_08

#### 30.8.2.2 Readout of the RX Buffer via ARDBn

The RX Buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- (1a): 32 Bit Access: Read QSPI\_ARDB0 <- 0x01\_02\_03\_04
- (2a): 32 Bit Access: Read QSPI\_ARDB1 <- 0x05\_06\_07\_08
- (1b/2b): 64 Bit Access: Read QSPI\_ARDB0 <- 0x01\_02\_03\_04\_05\_06\_07\_08

### 30.8.3 Reading Flash Data into the AHB Buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

This results in the AHB Buffer filled with:

**Table 30-38. Resulting AHB Buffer Content**

AHB Buffer Entry	Content
0	64'h01_02_03_04_05_06_07_08

#### 30.8.3.1 Readout of the AHB Buffer via Memory Mapped Read

The AHB Buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- (1a): 32 Bit Read Access: <- 0x01\_02\_03\_04
- (2a): 32 Bit Read Access: <- 0x05\_06\_07\_08
- (1/2): 64 Bit Read Access: <- 0x01\_02\_03\_04\_05\_06\_07\_08

## 30.9 Serial Flash Devices

Several different vendors make flash devices with a QuadSPI interface. At present there is no set standard for the QuadSPI instruction set. Most common commands currently have the same instruction code for all vendors, however some commands are unique to specific vendors. Some example sequences are provided below.

### 30.9.1 Example Sequences

This section provides the example sequences of the QuadSPI module.

**Table 30-39. Exit 4 x I/O Read Enhance Performance Mode (XIP) (Macronix) and Read Status**

INSTR	PAD	OPERAND	COMMENT
CMD	0x0	0xEB	4xI/O Read Command
ADDR	0x2	0x18	24 Bit address to be send on 4 pads
MODE	0x2	0x00	2 mode cycles (exit XIP)
DUMMY	0x0	0x04	4 dummy cycles
READ	0x2	0x08	Read 64 bits
CMD	0x0	0x05	Read Status register
READ	0x0	0x01	Status register data
STOP	0x0	0x00	STOP, Instruction over

#### 30.9.1.1 Fast Read Sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flashes.

**Table 30-40. Fast Read sequence**

Instruction	Pad	Operand	Comment
CMD	0x0	0x0B	Fast Read command = 0x0B
ADDR	0x0	0x18	24 Addr bits to be sent on one pad
DUMMY	0x0	0x08	8 Dummy cycles
READ	0x0	0x04	Read 32 Bits on one pad
JMP_ON_CS	0x0	0x00	Jump to instruction 0 (CMD)

#### 30.9.1.2 Fast Read Quad Output (Winbond)

The following table shows the Fast read quad output sequence for Winbond memories

**Table 30-41. Fast Read Quad output sequence**

Instruction	Pad	Operand	Comment
CMD	0x0	0x6B	Fast read quad output command = 0x6B

*Table continues on the next page...*

**Table 30-41. Fast Read Quad output sequence (continued)**

Instruction	Pad	Operand	Comment
ADDR	0x0	0x18	24 Addr bits to be sent on 1 pad
DUMMY	0x0	0x08	8 Dummy cycles
READ	0x2	0x04	Read 32 Bits on 4 pads
JMP_ON_CS	0x0	0x00	Jump to instruction 0 (CMD)

### 30.9.1.3 4 x I/O Read Enhance Performance Mode (XIP) (Macronix)

The following table shows the 4 x I/O Read Enhance Performance Mode for Macronix flashes. The enhanced performance mode is also known as XIP mode.

**Table 30-42. Fast Read Quad output sequence**

Instruction	Pad	Operand	Comment
CMD	0x0	0xEB	4xI/O Read command = 0xEB
ADDR	0x2	0x18	24 Addr bits to be sent on 4 pads
MODE	0x2	0xA5	2 mode cycles
DUMMY	0x0	0x04	4 Dummy cycles
READ	0x2	0x04	Read 32 Bits on 4 pads
JMP_ON_CS	0x0	0x01	Jump to instruction 1 (ADDR)

When in XIP mode the software should ensure that all the flashes connected to the controller are in XIP mode. As a part of initializing the controller, all the flashes may be enabled with XIP by carrying out dummy reads.

### 30.9.1.4 Dual Command Page Program (Numonyx)

The following table shows the Dual command page program sequence for Numonyx flashes.

**Table 30-43. Dual Command Page Program sequence**

Instruction	Pad	Operand	Comment
CMD	0x1	0x02	Dual command page program = 0x02 on 2 pads
ADDR	0x1	0x18	24 Addr bits to be sent on 2 pads
WRITE	0x1	0x20	Write 32 Bytes on 2 pads
STOP	0x0	0x00	STOP, Instruction over

### 30.9.1.5 Sector Erase (Macronix/Spansion/Numonyx)

The following table shows the Sector erase sequence for Macronix/Spansion/Numonyx flashes

**Table 30-44. Sector Erase sequence**

Instruction	Pad	Operand	Comment
CMD	0x0	0xD8	Sector erase command = 0xD8
ADDR	0x0	0x18	24 Addr bits to be sent on 1 pad
STOP	0x0	0x00	STOP, Instruction over

### 30.9.1.6 Read Status Register (Macronix/Spansion/Numonyx/Winbond)

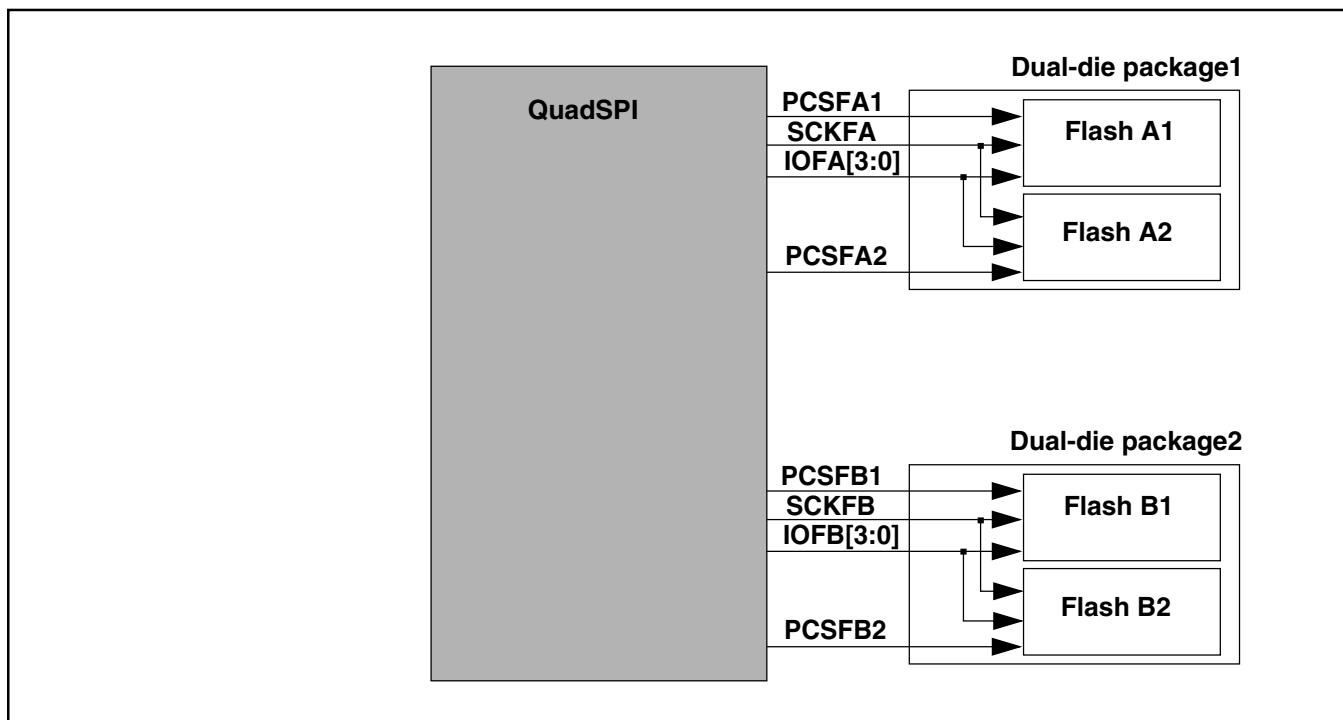
The following table shows the Read status register sequence for Macronix/Spansion/Numonyx/Winbond flashes.

**Table 30-45. Read Status Register Sequence**

Instruction	Pad	Operand	Comment
CMD	0x0	0x05	Read status register command = 0x05
READ	0x0	0x01	Read status register data
STOP	0x0	0x00	STOP, Instruction over

## 30.9.2 Dual Die Flashes

Certain serial flash vendors provide dual-die packages which are essentially two devices (dies) stacked within the same package to increase the memory capacity of a single package. These two devices within a package share the same data and clock pins, but have individual Chip Selects. QuadSPI controller provides support for two dual-die packages to be connected simultaneously. The figure below shows the two dual-die packages and the naming conventions used in this document. For simplicity, the data pins are shown to be unidirectional.



**Figure 30-8. Dual-die support**

Since the two devices within one package share the same i/o pads, they cannot function in parallel mode. Software should ensure that when QuadSPI is configured in parallel mode the two selected flash devices are from different dual-die packages.

### 30.9.3 Boot initialization sequence

The following are the recommended sequence of steps for booting from QuadSPI:

- System out of reset and flash available (300us)
- Clocks still at very low frequency. Clock tree configured, I/O pins configured. First request sent to QuadSPI for address 0x0 of flash.
- The reset command sequence in QuadSPI has 0x03 (basic read command) which is applicable to all flashes at < 50MHz serial flash clock
- The first few bytes of data is read from the flash which contains the following information:
  - The total sizes of all the flashes connected on board
  - Continuous mode entry sequence
  - 24bit or 32bit addressing (assuming 24bit for first accesses)

- All the serial flashes are configured
  - Quad Mode enabled
  - Dummy reads to enter into XIP
- QuadSPI is configured
  - Parallel enable set
  - LUT configured for highest performance reads
  - Buffers configured
- Serial flash clock frequency increased.
- Boot reads happen in parallel, quad output mode @66MHz.

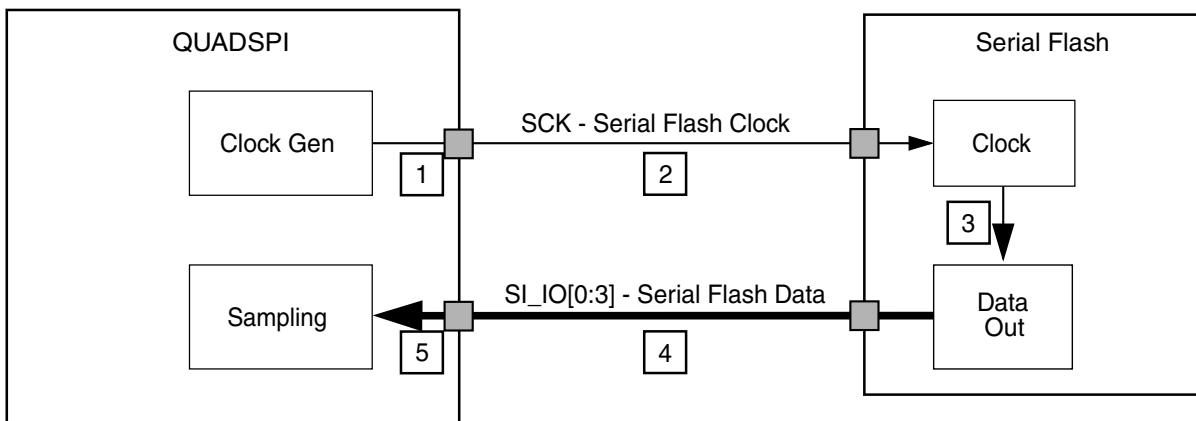
### 30.9.4 Serial Flash Clock Frequency Limitations

Certain commands of some serial flash devices are limited in the frequency applied to the serial flash device on command execution. In order to support these commands without having to recalibrate the module clocks, the the serial flash device clock can be divided by 2 (half speed) by setting the QSPI\_SMPR[HSENA] bit. The SCLK will return to full speed once the the QSPI\_SMPR[HSENA] bit is cleared.

## 30.10 Sampling of Serial Flash Input Data

### 30.10.1 Basic Description

QuadSPI is used to read data from the serial flash device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash device. Refer to the following figure for an overview of this scheme.

**Figure 30-9. Serial Flash Sampling Clock Overview**

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash. After a time of  $t_{Del,total}$  the data arrives at the internal sampling stage of the QuadSPI module. According to the Serial Flash Sampling Clock Overview figure, the following parts of the delay chain contribute to  $t_{Del,total}$ :

1. Output delay of the serial flash clock output of the device containing the QuadSPI module
2. Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash device
3. Clock to data out delay of the external serial flash device, including input and output delays
4. Wire delay of application/PCB from the external serial flash device to the device containing the QuadSPI module
5. Device delay corresponding to the input data

### **NOTE**

The amount of total delay  $t_{Del,total}$  is specific to the characteristics of the actual implementation. Also, the serial flash device clock (SCK) is inverted with respect to the QuadSPI internal reference clock.

## **30.10.2 SDR mode**

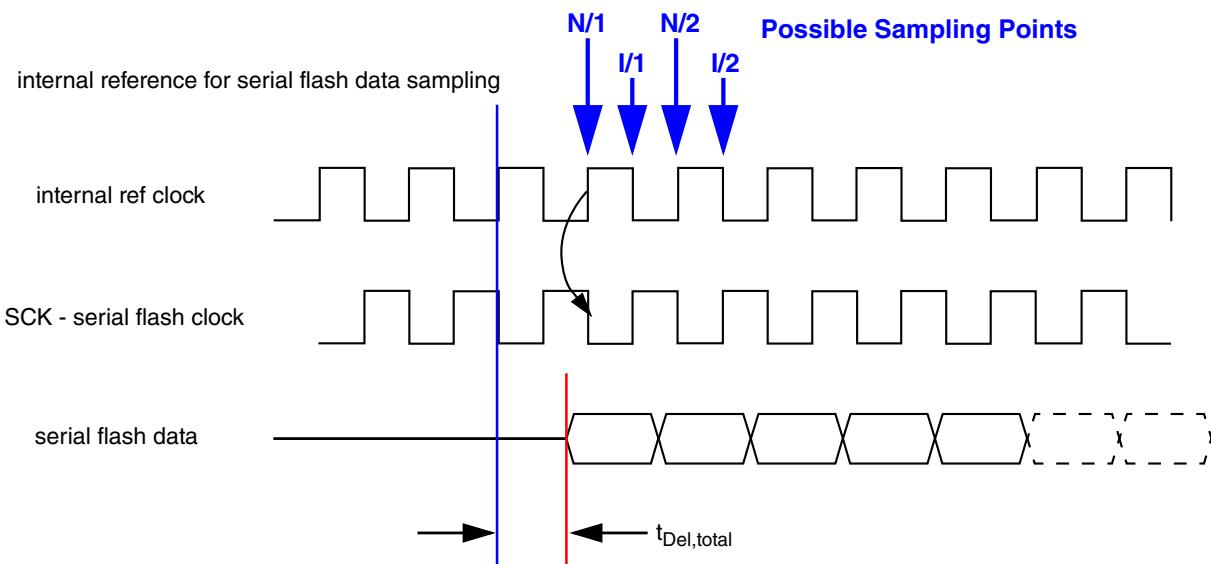
Most flash memories operate in single data rate (SDR) mode. In SDR mode, the data is transferred only on one edge of the clock signal. The SDR serial flash memories sample the incoming data on the rising edge of serial flash clock and drive the output data on the falling edge of the serial flash clock.

### 30.10.2.1 Internal sampling

**NOTE**

This mode may not be available on this chip. See the chip-specific QuadSPI information for the read modes that this chip supports.

QuadSPI uses different edges of the internal reference clock for sampling the input data in SDR mode.



**Figure 30-10. Internal sampling in SDR mode**

The possible points in time for sampling incoming data are denoted as N/1, I/1, N/2 and I/2 above. The sampling point relevant for the internal sampling is configured in the QSPI\_SMPR register. Refer to [Sampling Register \(QuadSPI\\_SMPR\)](#) for details. The following table gives an overview of the available configurations for the commands running at regular (full) speed:

**Table 30-46. Sampling Configuration**

Sampling Point	Description	Delay [FSDLY] [HSDLY]	Phase [FSPHS] [HSPHS]	QSPI_SMPR for Full Speed Setting <sup>1</sup>
N/1	sampling with non-inverted clock, 1 sample delay	0	0	0x0000000x
I/1	sampling with inverted clock, 1 sample delay	0	1	0x0000002x
N/2	sampling with non-inverted clock, 2 samples delay	1	0	0x0000004x
I/2	sampling with inverted clock, 2 samples delay	1	1	0x0000006x

1. 'x' is not considered here

Depending on the actual delay and the serial flash clock frequency, the appropriate sampling point can be chosen. The following remarks should be considered when selecting the appropriate setting:

- Theoretically there should be two settings possible to capture the correct data, since the serial flash output is valid for 1 clock cycle, disregarding rise and fall times and timing uncertainties.
- Depending on the timing uncertainties, it may turn out in actual applications that only one possible sample position remains. This is subject to careful consideration depending on the actual implementation.
- The delay  $t_{Del,total}$  is an absolute size to shift the point in time when the serial flash data get valid at the QuadSPI input.
- For decreasing frequency of the serial flash clock the distance between the edges increases. So for large differences in the frequency the required setting may change.
- For commands running at half of the regular serial flash clock (QSPI\_SMPR[HSENA] bit set) the sampling point must be figured separately to allow for the compensation of the absolute shift in time with respect to the sample-relative setting in the QSPI\_SPMR register.

### **30.10.2.2 DQS sampling method**

#### **NOTE**

This mode may not be available on this chip. See the chip-specific QuadSPI information for the read modes that this chip supports.

Data sampling in SDR mode can be supported using the DQS sampling method.

# **Chapter 31**

## **Queue Direct Memory Access Controller (qDMA)**

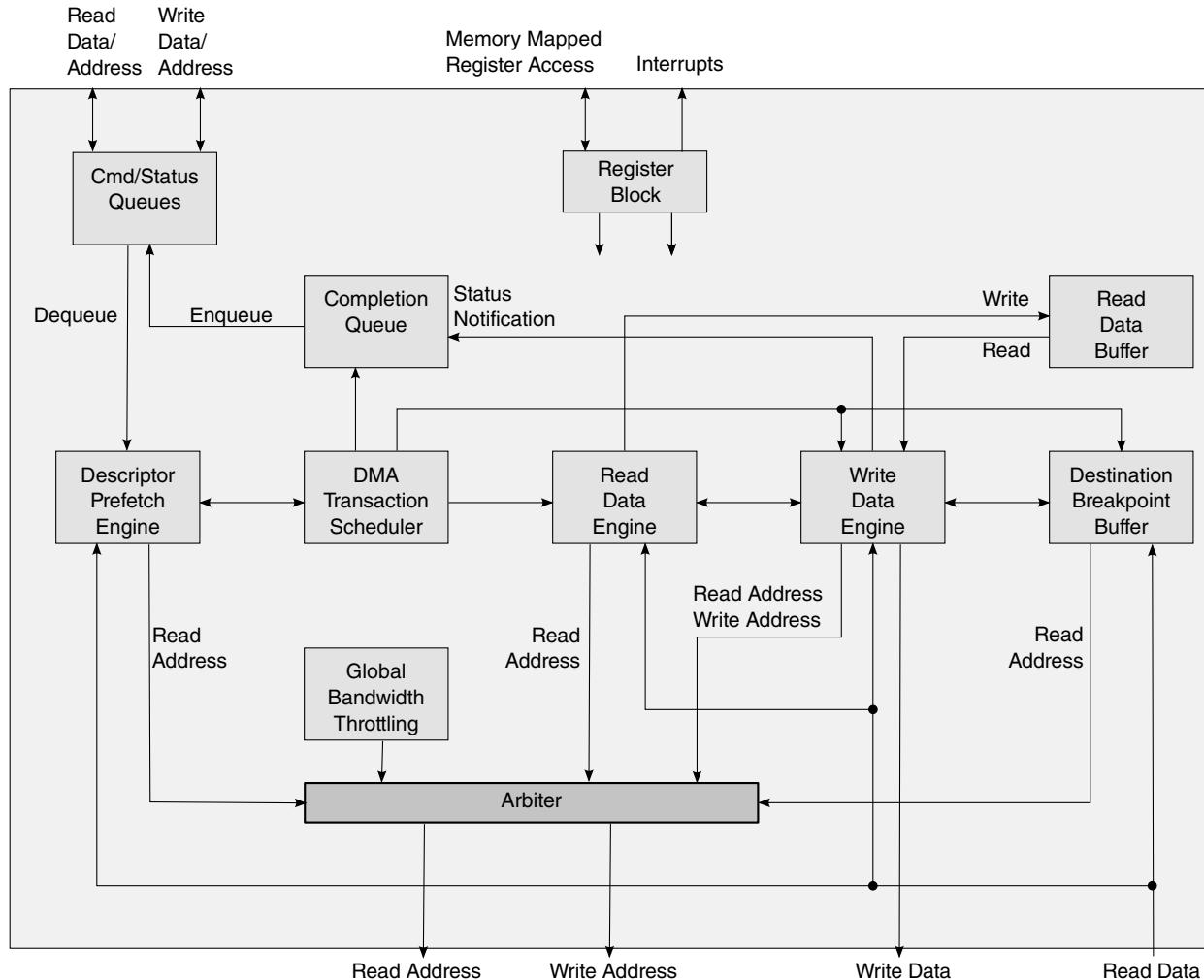
### **31.1 Introduction**

The qDMA controller transfers blocks of data between one source and one destination. The blocks of data transferred can be represented in memory as contiguous or non-contiguous using scatter/gather table(s). Channel virtualization is supported through enqueueing of DMA jobs to, or dequeuing DMA jobs from, different work queues.

#### **31.1.1 Overview**

The qDMA supports channel virtualization by allowing DMA jobs to be enqueued into different command queues. Core can initiate a DMA transaction by preparing a command descriptor (CD) for each DMA job and enqueueing this job to a command queue. The qDMA prefetches DMA jobs from the command queues. It then schedules and dispatches to internal DMA hardware engines, which generate read and write requests. Both qDMA source data and destination data can be either contiguous or non-contiguous using one or more scatter/gather tables.

The qDMA supports global bandwidth flow control where all DMA transactions are stalled if the bandwidth threshold has been reached. Also supported are transaction based read and write throttling.

**Figure 31-1. qDMA Block Diagram**

### 31.1.2 Features

The queue DMA controller (qDMA) has the following features:

- Supports command descriptor prefetch.
  - Supports single dequeue from command queues, store up to 4 prefetched command descriptors.
- Supports channel virtualization.
  - Allows DMA jobs to be enqueued into different status queues. Core can initiate DMA transaction by preparing command descriptor for each DMA job and enqueueing this job to a command queue.
- Supports the following command queue features

- Supports multiple virtualized blocks for multi-core support.
- Supports 8 command queues per virtualized block.
- Supports 1 status queue per virtualized block.
- Supports 64 to 16384 command descriptor entries per queue.
- Supports "transaction error", "programming error", and "coalescing" (command and timer threshold) interrupts.
- Supports transaction based destination stride and source stride.
  - Destination stride attributes and source stride attributes can be setup in corresponding source descriptor and destination descriptors
- Supports destination and source address hold function.
- Supports transaction based cache attribute setup.
  - Cache attributes can be setup in corresponding frame descriptor and destination descriptors
- Supports transaction based QoS.
  - Source read and destination write QoS can be setup in corresponding command descriptor and source/destination descriptor(s).
- Supports destination based throttle control.
  - Throttle control per destination can be setup through destination descriptor.
- Supports DMA transfers with misaligned addresses and any transaction size.
- Supports global DMA bandwidth control where transaction generated by the DMA is stalled when the system bandwidth consumption threshold allocated to the DMA has been reached.
- Supports source/destination data scatter/gather through S/G table located in the system memory.
  - Source entry and each destination entry has an extension bit to indicate if the corresponding data block is contiguous or scattered/gathered.
- Support partition reset assistance.
  - Privileged software can flush jobs based on logical I/O device partition.

### 31.1.3 Modes of Operation

qDMA only supports command queue mode.

The command queue mode provides core off-loading and channel virtualization through the memory block assignment.

## 31.2 Memory Map

### NOTE

Systems may have software which runs at different privilege levels. In this document the term privileged or privileged software refers to the most privileged software layer, which could be an operating system or software hypervisor.

This section provides a detailed description of all accessible qDMA memory and registers. qDMA uses two contiguous 64KB memory pages for isolation of privileged versus manager registers as shown in [Table 31-1](#). The first page is intended for privileged software for setup and control of the qDMA hardware resources. The second page allows management software to operate the qDMA.

**Table 31-1. qDMA Contiguous Address Map within CCSR Space**

Address Offset	Registers
0x0_0000-0x0_FFFF	Privileged registers
0x1_0000-0x1_FFFF	Manager registers

The table here shows the virtualized 64KB pages utilized by the qDMA command queues for multi-core support.

**Table 31-2. qDMA Command/Status Queue Mapping in CCSR Space**

Address Offset	Registers	Block
0x2_0nC0-0x2_0nFF	QMLite Command Queue 0-7	0
0x2_0800-0x2_08FF	QMLite Status Queue	
0x2_0A00-0x2_0AFF	QMLite Queue Global Registers	
0x3_0nC0-0x3_0nFF	QMLite Command Queue 0-7	1
0x3_0800-0x3_08FF	QMLite Status Queue	
0x4_0nC0-0x4_0nFF	QMLite Command Queue 0-7	2
0x4_0800-0x4_08FF	QMLite Status Queue	
0x5_0nC0-0x5_0nFF	QMLite Command Queue 0-7	3
0x5_0800-0x5_08FF	QMLite Status Queue	

### 31.2.1 qDMA register descriptions

The table shows the memory map for management of the qDMA resources.

### 31.2.1.1 qdma memory map

qDMA base address: 838\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA mode register (DMR)	32	RW	0000_0000h
4h	DMA status register (DSR_P)	32	RO	0000_0000h
40h	DMA stream ID flush register (DSIDFR)	32	RW	0000_0000h
1_0004h	DMA status register (DSR_M)	32	RO	0000_0000h
1_0040h	DMA global bandwidth throttle register (DGBTTR)	32	RW	0008_0000h
1_0BF8h	IP Block Revision register 0 (IPBRR0)	32	RO	0A30_0100h
1_0BFCh	IP Block Revision register 1 (IPBRR1)	32	RO	0000_0000h
1_0E00h	DMA error interrupt enable register (DEIER)	32	RW	0000_0000h
1_0E04h	DMA error detect register (DEDR)	32	W1C	0000_0000h
1_0E10h	DMA error capture frame descriptor word 0 register (DECCDW0R)	32	RO	0000_0000h
1_0E14h	DMA error capture frame descriptor word 1 register (DECCDW1R)	32	RO	0000_0000h
1_0E18h	DMA error capture frame descriptor word 2 register (DECCDW2R)	32	RO	0000_0000h
1_0E1Ch	DMA error capture frame descriptor word 3 register (DECCDW3R)	32	RO	0000_0000h
1_0E30h	DMA error capture command queue ID register (DECCQIDR)	32	RO	0000_0000h
1_0E34h	DMA error capture byte count register (DECBR)	32	RO	0000_0000h
2_00C0h	Block 0 command queue 0 mode register (B0CQ0MR)	32	RW	0000_0000h
2_00C4h	Block 0 command queue 0 status register (B0CQ0SR)	32	RO	0000_0000h
2_00C8h	Block 0 command queue 0 extended dequeue pointer address register (B0CQ0EDPAR)	32	RW	0000_0000h
2_00CCh	Block 0 command queue 0 dequeue pointer address register (B0CQ0DPAR)	32	RW	0000_0000h
2_00D0h	Block 0 command queue 0 extended enqueue pointer address register (B0CQ0EEPAR)	32	RW	0000_0000h
2_00D4h	Block 0 command queue 0 enqueue pointer address register (B0CQ0EPAR)	32	RW	0000_0000h
2_00E0h	Block 0 command queue interrupt enable registers (B0CQIER)	32	RW	0000_0000h
2_00E4h	Block 0 command queue interrupt detect registers (B0CQIDR)	32	W1C	0000_0000h
2_01C0h	Block 0 command queue 1 mode register (B0CQ1MR)	32	RW	0000_0000h
2_01C4h	Block 0 command queue 1 status register (B0CQ1SR)	32	RO	0000_0000h
2_01C8h	Block 0 command queue 1 extended dequeue pointer address register (B0CQ1EDPAR)	32	RW	0000_0000h
2_01CCh	Block 0 command queue 1 dequeue pointer address register (B0CQ1DPAR)	32	RW	0000_0000h
2_01D0h	Block 0 command queue 1 extended enqueue pointer address register (B0CQ1EEPAR)	32	RW	0000_0000h
2_01D4h	Block 0 command queue 1 enqueue pointer address register (B0CQ1EPAR)	32	RW	0000_0000h

Table continues on the next page...

## Memory Map

Offset	Register	Width (In bits)	Access	Reset value
2_02C0h	Block 0 command queue 2 mode register (B0CQ2MR)	32	RW	0000_0000h
2_02C4h	Block 0 command queue 2 status register (B0CQ2SR)	32	RO	0000_0000h
2_02C8h	Block 0 command queue 2 extended dequeue pointer address register (B0CQ2EDPAR)	32	RW	0000_0000h
2_02CCh	Block 0 command queue 2 dequeue pointer address register (B0CQ2DPAR)	32	RW	0000_0000h
2_02D0h	Block 0 command queue 2 extended enqueue pointer address register (B0CQ2EPPAR)	32	RW	0000_0000h
2_02D4h	Block 0 command queue 2 enqueue pointer address register (B0CQ2EPAR)	32	RW	0000_0000h
2_03C0h	Block 0 command queue 3 mode register (B0CQ3MR)	32	RW	0000_0000h
2_03C4h	Block 0 command queue 3 status register (B0CQ3SR)	32	RO	0000_0000h
2_03C8h	Block 0 command queue 3 extended dequeue pointer address register (B0CQ3EDPAR)	32	RW	0000_0000h
2_03CCh	Block 0 command queue 3 dequeue pointer address register (B0CQ3DPAR)	32	RW	0000_0000h
2_03D0h	Block 0 command queue 3 extended enqueue pointer address register (B0CQ3EPPAR)	32	RW	0000_0000h
2_03D4h	Block 0 command queue 3 enqueue pointer address register (B0CQ3EPAR)	32	RW	0000_0000h
2_04C0h	Block 0 command queue 4 mode register (B0CQ4MR)	32	RW	0000_0000h
2_04C4h	Block 0 command queue 4 status register (B0CQ4SR)	32	RO	0000_0000h
2_04C8h	Block 0 command queue 4 extended dequeue pointer address register (B0CQ4EDPAR)	32	RW	0000_0000h
2_04CCh	Block 0 command queue 4 dequeue pointer address register (B0CQ4DPAR)	32	RW	0000_0000h
2_04D0h	Block 0 command queue 4 extended enqueue pointer address register (B0CQ4EPPAR)	32	RW	0000_0000h
2_04D4h	Block 0 command queue 4 enqueue pointer address register (B0CQ4EPAR)	32	RW	0000_0000h
2_05C0h	Block 0 command queue 5 mode register (B0CQ5MR)	32	RW	0000_0000h
2_05C4h	Block 0 command queue 5 status register (B0CQ5SR)	32	RO	0000_0000h
2_05C8h	Block 0 command queue 5 extended dequeue pointer address register (B0CQ5EDPAR)	32	RW	0000_0000h
2_05CCh	Block 0 command queue 5 dequeue pointer address register (B0CQ5DPAR)	32	RW	0000_0000h
2_05D0h	Block 0 command queue 5 extended enqueue pointer address register (B0CQ5EPPAR)	32	RW	0000_0000h
2_05D4h	Block 0 command queue 5 enqueue pointer address register (B0CQ5EPAR)	32	RW	0000_0000h
2_06C0h	Block 0 command queue 6 mode register (B0CQ6MR)	32	RW	0000_0000h
2_06C4h	Block 0 command queue 6 status register (B0CQ6SR)	32	RO	0000_0000h
2_06C8h	Block 0 command queue 6 extended dequeue pointer address register (B0CQ6EDPAR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2_06CCh	Block 0 command queue 6 dequeue pointer address register (B0CQ6DPAR)	32	RW	0000_0000h
2_06D0h	Block 0 command queue 6 extended enqueue pointer address register (B0CQ6EPPAR)	32	RW	0000_0000h
2_06D4h	Block 0 command queue 6 enqueue pointer address register (B0CQ6EPAR)	32	RW	0000_0000h
2_07C0h	Block 0 command queue 7 mode register (B0CQ7MR)	32	RW	0000_0000h
2_07C4h	Block 0 command queue 7 status register (B0CQ7SR)	32	RO	0000_0000h
2_07C8h	Block 0 command queue 7 extended dequeue pointer address register (B0CQ7EDPAR)	32	RW	0000_0000h
2_07CCh	Block 0 command queue 7 dequeue pointer address register (B0CQ7DPAR)	32	RW	0000_0000h
2_07D0h	Block 0 command queue 7 extended enqueue pointer address register (B0CQ7EPPAR)	32	RW	0000_0000h
2_07D4h	Block 0 command queue 7 enqueue pointer address register (B0CQ7EPAR)	32	RW	0000_0000h
2_0800h	Block 0 status queue mode register (B0SQMR)	32	RW	0000_0000h
2_0804h	Block 0 status queue status register (B0SQSR)	32	RO	0000_0000h
2_0808h	Block 0 status queue extended dequeue pointer address register (B0SQEDPAR)	32	RW	0000_0000h
2_080Ch	Block 0 status queue dequeue pointer address register (B0SQDPAR)	32	RW	0000_0000h
2_0810h	Block 0 status queue extended enqueue pointer address register (B0SQEPPAR)	32	RW	0000_0000h
2_0814h	Block 0 status queue enqueue pointer address register (B0SQEPAR)	32	RW	0000_0000h
2_0828h	Block 0 status queue interrupt coalescing register (B0SQICR)	32	RW	0000_0000h
2_0A00h	Command queue mode register (CQMR)	32	RW	0000_0000h
2_0A08h	Command queue dequeue scheduler configuration register 1 (CQDS CR1)	32	RW	0123_4567h
2_0A0Ch	Command queue dequeue scheduler configuration register 2 (CQDS CR2)	32	RW	0000_0000h
2_0A10h	Command queue interrupt enable register (CQIER)	32	RW	0000_0000h
2_0A14h	Command queue error detect register (CQEDR)	32	W1C	0000_0000h
2_0A18h	Command queue error capture extended address register (CQEC EAR)	32	RO	0000_0000h
2_0A1Ch	Command Queue Error Capture Address Register (CQECAr)	32	RO	0000_0000h
2_0A20h	Status queue critical congestion management register (SQCCMR)	32	RW	0000_0000h
2_0A80h - 2_0A8Ch	Command queue block a stream ID register (CQ0SIDR - CQ3SIDR)	32	RW	0000_0000h
3_00C0h	Block 1 command queue 0 mode register (B1CQ0MR)	32	RW	0000_0000h
3_00C4h	Block 1 command queue 0 status register (B1CQ0SR)	32	RO	0000_0000h
3_00C8h	Block 1 command queue 0 extended dequeue pointer address register (B1CQ0EDPAR)	32	RW	0000_0000h
3_00CCh	Block 1 command queue 0 dequeue pointer address register (B1CQ0DPAR)	32	RW	0000_0000h

Table continues on the next page...

## Memory Map

Offset	Register	Width (In bits)	Access	Reset value
3_00D0h	Block 1 command queue 0 extended enqueue pointer address register (B1CQ0EEPAR)	32	RW	0000_0000h
3_00D4h	Block 1 command queue 0 enqueue pointer address register (B1CQ0EPAR)	32	RW	0000_0000h
3_00E0h	Block 1 command queue interrupt enable registers (B1CQIER)	32	RW	0000_0000h
3_00E4h	Block 1 command queue interrupt detect registers (B1CQIDR)	32	W1C	0000_0000h
3_01C0h	Block 1 command queue 1 mode register (B1CQ1MR)	32	RW	0000_0000h
3_01C4h	Block 1 command queue 1 status register (B1CQ1SR)	32	RO	0000_0000h
3_01C8h	Block 1 command queue 1 extended dequeue pointer address register (B1CQ1EDPAR)	32	RW	0000_0000h
3_01CCh	Block 1 command queue 1 dequeue pointer address register (B1CQ1DPAR)	32	RW	0000_0000h
3_01D0h	Block 1 command queue 1 extended enqueue pointer address register (B1CQ1EEPAR)	32	RW	0000_0000h
3_01D4h	Block 1 command queue 1 enqueue pointer address register (B1CQ1EPAR)	32	RW	0000_0000h
3_02C0h	Block 1 command queue 2 mode register (B1CQ2MR)	32	RW	0000_0000h
3_02C4h	Block 1 command queue 2 status register (B1CQ2SR)	32	RO	0000_0000h
3_02C8h	Block 1 command queue 2 extended dequeue pointer address register (B1CQ2EDPAR)	32	RW	0000_0000h
3_02CCh	Block 1 command queue 2 dequeue pointer address register (B1CQ2DPAR)	32	RW	0000_0000h
3_02D0h	Block 1 command queue 2 extended enqueue pointer address register (B1CQ2EEPAR)	32	RW	0000_0000h
3_02D4h	Block 1 command queue 2 enqueue pointer address register (B1CQ2EPAR)	32	RW	0000_0000h
3_03C0h	Block 1 command queue 3 mode register (B1CQ3MR)	32	RW	0000_0000h
3_03C4h	Block 1 command queue 3 status register (B1CQ3SR)	32	RO	0000_0000h
3_03C8h	Block 1 command queue 3 extended dequeue pointer address register (B1CQ3EDPAR)	32	RW	0000_0000h
3_03CCh	Block 1 command queue 3 dequeue pointer address register (B1CQ3DPAR)	32	RW	0000_0000h
3_03D0h	Block 1 command queue 3 extended enqueue pointer address register (B1CQ3EEPAR)	32	RW	0000_0000h
3_03D4h	Block 1 command queue 3 enqueue pointer address register (B1CQ3EPAR)	32	RW	0000_0000h
3_04C0h	Block 1 command queue 4 mode register (B1CQ4MR)	32	RW	0000_0000h
3_04C4h	Block 1 command queue 4 status register (B1CQ4SR)	32	RO	0000_0000h
3_04C8h	Block 1 command queue 4 extended dequeue pointer address register (B1CQ4EDPAR)	32	RW	0000_0000h
3_04CCh	Block 1 command queue 4 dequeue pointer address register (B1CQ4DPAR)	32	RW	0000_0000h
3_04D0h	Block 1 command queue 4 extended enqueue pointer address register (B1CQ4EEPAR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
3_04D4h	Block 1 command queue 4 enqueue pointer address register (B1CQ4EPAR)	32	RW	0000_0000h
3_05C0h	Block 1 command queue 5 mode register (B1CQ5MR)	32	RW	0000_0000h
3_05C4h	Block 1 command queue 5 status register (B1CQ5SR)	32	RO	0000_0000h
3_05C8h	Block 1 command queue 5 extended dequeue pointer address register (B1CQ5EDPAR)	32	RW	0000_0000h
3_05CCh	Block 1 command queue 5 dequeue pointer address register (B1CQ5DPAR)	32	RW	0000_0000h
3_05D0h	Block 1 command queue 5 extended enqueue pointer address register (B1CQ5EEPAR)	32	RW	0000_0000h
3_05D4h	Block 1 command queue 5 enqueue pointer address register (B1CQ5EPAR)	32	RW	0000_0000h
3_06C0h	Block 1 command queue 6 mode register (B1CQ6MR)	32	RW	0000_0000h
3_06C4h	Block 1 command queue 6 status register (B1CQ6SR)	32	RO	0000_0000h
3_06C8h	Block 1 command queue 6 extended dequeue pointer address register (B1CQ6EDPAR)	32	RW	0000_0000h
3_06CCh	Block 1 command queue 6 dequeue pointer address register (B1CQ6DPAR)	32	RW	0000_0000h
3_06D0h	Block 1 command queue 6 extended enqueue pointer address register (B1CQ6EEPAR)	32	RW	0000_0000h
3_06D4h	Block 1 command queue 6 enqueue pointer address register (B1CQ6EPAR)	32	RW	0000_0000h
3_07C0h	Block 1 command queue 7 mode register (B1CQ7MR)	32	RW	0000_0000h
3_07C4h	Block 1 command queue 7 status register (B1CQ7SR)	32	RO	0000_0000h
3_07C8h	Block 1 command queue 7 extended dequeue pointer address register (B1CQ7EDPAR)	32	RW	0000_0000h
3_07CCh	Block 1 command queue 7 dequeue pointer address register (B1CQ7DPAR)	32	RW	0000_0000h
3_07D0h	Block 1 command queue 7 extended enqueue pointer address register (B1CQ7EEPAR)	32	RW	0000_0000h
3_07D4h	Block 1 command queue 7 enqueue pointer address register (B1CQ7EPAR)	32	RW	0000_0000h
3_0800h	Block 1 status queue mode register (B1SQMR)	32	RW	0000_0000h
3_0804h	Block 1 status queue status register (B1SQSR)	32	RO	0000_0000h
3_0808h	Block 1 status queue extended dequeue pointer address register (B1SQEDPAR)	32	RW	0000_0000h
3_080Ch	Block 1 status queue dequeue pointer address register (B1SQDPAR)	32	RW	0000_0000h
3_0810h	Block 1 status queue extended enqueue pointer address register (B1SQEEPAR)	32	RW	0000_0000h
3_0814h	Block 1 status queue enqueue pointer address register (B1SQEPAR)	32	RW	0000_0000h
3_0828h	Block 1 status queue interrupt coalescing register (B1SQICR)	32	RW	0000_0000h
4_00C0h	Block 2 command queue 0 mode register (B2CQ0MR)	32	RW	0000_0000h
4_00C4h	Block 2 command queue 0 status register (B2CQ0SR)	32	RO	0000_0000h

Table continues on the next page...

## Memory Map

Offset	Register	Width (In bits)	Access	Reset value
4_00C8h	Block 2 command queue 0 extended dequeue pointer address register (B2CQ0EDPAR)	32	RW	0000_0000h
4_00CCh	Block 2 command queue 0 dequeue pointer address register (B2CQ0DPAR)	32	RW	0000_0000h
4_00D0h	Block 2 command queue 0 extended enqueue pointer address register (B2CQ0EEPAR)	32	RW	0000_0000h
4_00D4h	Block 2 command queue 0 enqueue pointer address register (B2CQ0EPAR)	32	RW	0000_0000h
4_00E0h	Block 2 command queue interrupt enable registers (B2CQIER)	32	RW	0000_0000h
4_00E4h	Block 2 command queue interrupt detect registers (B2CQIDR)	32	W1C	0000_0000h
4_01C0h	Block 2 command queue 1 mode register (B2CQ1MR)	32	RW	0000_0000h
4_01C4h	Block 2 command queue 1 status register (B2CQ1SR)	32	RO	0000_0000h
4_01C8h	Block 2 command queue 1 extended dequeue pointer address register (B2CQ1EDPAR)	32	RW	0000_0000h
4_01CCh	Block 2 command queue 1 dequeue pointer address register (B2CQ1DPAR)	32	RW	0000_0000h
4_01D0h	Block 2 command queue 1 extended enqueue pointer address register (B2CQ1EEPAR)	32	RW	0000_0000h
4_01D4h	Block 2 command queue 1 enqueue pointer address register (B2CQ1EPAR)	32	RW	0000_0000h
4_02C0h	Block 2 command queue 2 mode register (B2CQ2MR)	32	RW	0000_0000h
4_02C4h	Block 2 command queue 2 status register (B2CQ2SR)	32	RO	0000_0000h
4_02C8h	Block 2 command queue 2 extended dequeue pointer address register (B2CQ2EDPAR)	32	RW	0000_0000h
4_02CCh	Block 2 command queue 2 dequeue pointer address register (B2CQ2DPAR)	32	RW	0000_0000h
4_02D0h	Block 2 command queue 2 extended enqueue pointer address register (B2CQ2EEPAR)	32	RW	0000_0000h
4_02D4h	Block 2 command queue 2 enqueue pointer address register (B2CQ2EPAR)	32	RW	0000_0000h
4_03C0h	Block 2 command queue 3 mode register (B2CQ3MR)	32	RW	0000_0000h
4_03C4h	Block 2 command queue 3 status register (B2CQ3SR)	32	RO	0000_0000h
4_03C8h	Block 2 command queue 3 extended dequeue pointer address register (B2CQ3EDPAR)	32	RW	0000_0000h
4_03CCh	Block 2 command queue 3 dequeue pointer address register (B2CQ3DPAR)	32	RW	0000_0000h
4_03D0h	Block 2 command queue 3 extended enqueue pointer address register (B2CQ3EEPAR)	32	RW	0000_0000h
4_03D4h	Block 2 command queue 3 enqueue pointer address register (B2CQ3EPAR)	32	RW	0000_0000h
4_04C0h	Block 2 command queue 4 mode register (B2CQ4MR)	32	RW	0000_0000h
4_04C4h	Block 2 command queue 4 status register (B2CQ4SR)	32	RO	0000_0000h
4_04C8h	Block 2 command queue 4 extended dequeue pointer address register (B2CQ4EDPAR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
4_04CCh	Block 2 command queue 4 dequeue pointer address register (B2CQ4DPAR)	32	RW	0000_0000h
4_04D0h	Block 2 command queue 4 extended enqueue pointer address register (B2CQ4EEPAR)	32	RW	0000_0000h
4_04D4h	Block 2 command queue 4 enqueue pointer address register (B2CQ4EPAR)	32	RW	0000_0000h
4_05C0h	Block 2 command queue 5 mode register (B2CQ5MR)	32	RW	0000_0000h
4_05C4h	Block 2 command queue 5 status register (B2CQ5SR)	32	RO	0000_0000h
4_05C8h	Block 2 command queue 5 extended dequeue pointer address register (B2CQ5EDPAR)	32	RW	0000_0000h
4_05CCh	Block 2 command queue 5 dequeue pointer address register (B2CQ5DPAR)	32	RW	0000_0000h
4_05D0h	Block 2 command queue 5 extended enqueue pointer address register (B2CQ5EEPAR)	32	RW	0000_0000h
4_05D4h	Block 2 command queue 5 enqueue pointer address register (B2CQ5EPAR)	32	RW	0000_0000h
4_06C0h	Block 2 command queue 6 mode register (B2CQ6MR)	32	RW	0000_0000h
4_06C4h	Block 2 command queue 6 status register (B2CQ6SR)	32	RO	0000_0000h
4_06C8h	Block 2 command queue 6 extended dequeue pointer address register (B2CQ6EDPAR)	32	RW	0000_0000h
4_06CCh	Block 2 command queue 6 dequeue pointer address register (B2CQ6DPAR)	32	RW	0000_0000h
4_06D0h	Block 2 command queue 6 extended enqueue pointer address register (B2CQ6EEPAR)	32	RW	0000_0000h
4_06D4h	Block 2 command queue 6 enqueue pointer address register (B2CQ6EPAR)	32	RW	0000_0000h
4_07C0h	Block 2 command queue 7 mode register (B2CQ7MR)	32	RW	0000_0000h
4_07C4h	Block 2 command queue 7 status register (B2CQ7SR)	32	RO	0000_0000h
4_07C8h	Block 2 command queue 7 extended dequeue pointer address register (B2CQ7EDPAR)	32	RW	0000_0000h
4_07CCh	Block 2 command queue 7 dequeue pointer address register (B2CQ7DPAR)	32	RW	0000_0000h
4_07D0h	Block 2 command queue 7 extended enqueue pointer address register (B2CQ7EEPAR)	32	RW	0000_0000h
4_07D4h	Block 2 command queue 7 enqueue pointer address register (B2CQ7EPAR)	32	RW	0000_0000h
4_0800h	Block 2 status queue mode register (B2SQMR)	32	RW	0000_0000h
4_0804h	Block 2 status queue status register (B2SQSR)	32	RO	0000_0000h
4_0808h	Block 2 status queue extended dequeue pointer address register (B2SQEDPAR)	32	RW	0000_0000h
4_080Ch	Block 2 status queue dequeue pointer address register (B2SQDPAR)	32	RW	0000_0000h
4_0810h	Block 2 status queue extended enqueue pointer address register (B2SQEEPAR)	32	RW	0000_0000h
4_0814h	Block 2 status queue enqueue pointer address register (B2SQEPAR)	32	RW	0000_0000h

Table continues on the next page...

## Memory Map

Offset	Register	Width (In bits)	Access	Reset value
4_0828h	Block 2 status queue interrupt coalescing register (B2SQICR)	32	RW	0000_0000h
5_00C0h	Block 3 command queue 0 mode register (B3CQ0MR)	32	RW	0000_0000h
5_00C4h	Block 3 command queue 0 status register (B3CQ0SR)	32	RO	0000_0000h
5_00C8h	Block 3 command queue 0 extended dequeue pointer address register (B3CQ0EDPAR)	32	RW	0000_0000h
5_00CCh	Block 3 command queue 0 dequeue pointer address register (B3CQ0DPAR)	32	RW	0000_0000h
5_00D0h	Block 3 command queue 0 extended enqueue pointer address register (B3CQ0EEPAR)	32	RW	0000_0000h
5_00D4h	Block 3 command queue 0 enqueue pointer address register (B3CQ0EPAR)	32	RW	0000_0000h
5_00E0h	Block 3 command queue interrupt enable registers (B3CQIER)	32	RW	0000_0000h
5_00E4h	Block 3 command queue interrupt detect registers (B3CQIDR)	32	W1C	0000_0000h
5_01C0h	Block 3 command queue 1 mode register (B3CQ1MR)	32	RW	0000_0000h
5_01C4h	Block 3 command queue 1 status register (B3CQ1SR)	32	RO	0000_0000h
5_01C8h	Block 3 command queue 1 extended dequeue pointer address register (B3CQ1EDPAR)	32	RW	0000_0000h
5_01CCh	Block 3 command queue 1 dequeue pointer address register (B3CQ1DPAR)	32	RW	0000_0000h
5_01D0h	Block 3 command queue 1 extended enqueue pointer address register (B3CQ1EEPAR)	32	RW	0000_0000h
5_01D4h	Block 3 command queue 1 enqueue pointer address register (B3CQ1EPAR)	32	RW	0000_0000h
5_02C0h	Block 3 command queue 2 mode register (B3CQ2MR)	32	RW	0000_0000h
5_02C4h	Block 3 command queue 2 status register (B3CQ2SR)	32	RO	0000_0000h
5_02C8h	Block 3 command queue 2 extended dequeue pointer address register (B3CQ2EDPAR)	32	RW	0000_0000h
5_02CCh	Block 3 command queue 2 dequeue pointer address register (B3CQ2DPAR)	32	RW	0000_0000h
5_02D0h	Block 3 command queue 2 extended enqueue pointer address register (B3CQ2EEPAR)	32	RW	0000_0000h
5_02D4h	Block 3 command queue 2 enqueue pointer address register (B3CQ2EPAR)	32	RW	0000_0000h
5_03C0h	Block 3 command queue 3 mode register (B3CQ3MR)	32	RW	0000_0000h
5_03C4h	Block 3 command queue 3 status register (B3CQ3SR)	32	RO	0000_0000h
5_03C8h	Block 3 command queue 3 extended dequeue pointer address register (B3CQ3EDPAR)	32	RW	0000_0000h
5_03CCh	Block 3 command queue 3 dequeue pointer address register (B3CQ3DPAR)	32	RW	0000_0000h
5_03D0h	Block 3 command queue 3 extended enqueue pointer address register (B3CQ3EEPAR)	32	RW	0000_0000h
5_03D4h	Block 3 command queue 3 enqueue pointer address register (B3CQ3EPAR)	32	RW	0000_0000h
5_04C0h	Block 3 command queue 4 mode register (B3CQ4MR)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
5_04C4h	Block 3 command queue 4 status register (B3CQ4SR)	32	RO	0000_0000h
5_04C8h	Block 3 command queue 4 extended dequeue pointer address register (B3CQ4EDPAR)	32	RW	0000_0000h
5_04CCh	Block 3 command queue 4 dequeue pointer address register (B3CQ4DPAR)	32	RW	0000_0000h
5_04D0h	Block 3 command queue 4 extended enqueue pointer address register (B3CQ4EEPAR)	32	RW	0000_0000h
5_04D4h	Block 3 command queue 4 enqueue pointer address register (B3CQ4EPAR)	32	RW	0000_0000h
5_05C0h	Block 3 command queue 5 mode register (B3CQ5MR)	32	RW	0000_0000h
5_05C4h	Block 3 command queue 5 status register (B3CQ5SR)	32	RO	0000_0000h
5_05C8h	Block 3 command queue 5 extended dequeue pointer address register (B3CQ5EDPAR)	32	RW	0000_0000h
5_05CCh	Block 3 command queue 5 dequeue pointer address register (B3CQ5DPAR)	32	RW	0000_0000h
5_05D0h	Block 3 command queue 5 extended enqueue pointer address register (B3CQ5EEPAR)	32	RW	0000_0000h
5_05D4h	Block 3 command queue 5 enqueue pointer address register (B3CQ5EPAR)	32	RW	0000_0000h
5_06C0h	Block 3 command queue 6 mode register (B3CQ6MR)	32	RW	0000_0000h
5_06C4h	Block 3 command queue 6 status register (B3CQ6SR)	32	RO	0000_0000h
5_06C8h	Block 3 command queue 6 extended dequeue pointer address register (B3CQ6EDPAR)	32	RW	0000_0000h
5_06CCh	Block 3 command queue 6 dequeue pointer address register (B3CQ6DPAR)	32	RW	0000_0000h
5_06D0h	Block 3 command queue 6 extended enqueue pointer address register (B3CQ6EEPAR)	32	RW	0000_0000h
5_06D4h	Block 3 command queue 6 enqueue pointer address register (B3CQ6EPAR)	32	RW	0000_0000h
5_07C0h	Block 3 command queue 7 mode register (B3CQ7MR)	32	RW	0000_0000h
5_07C4h	Block 3 command queue 7 status register (B3CQ7SR)	32	RO	0000_0000h
5_07C8h	Block 3 command queue 7 extended dequeue pointer address register (B3CQ7EDPAR)	32	RW	0000_0000h
5_07CCh	Block 3 command queue 7 dequeue pointer address register (B3CQ7DPAR)	32	RW	0000_0000h
5_07D0h	Block 3 command queue 7 extended enqueue pointer address register (B3CQ7EEPAR)	32	RW	0000_0000h
5_07D4h	Block 3 command queue 7 enqueue pointer address register (B3CQ7EPAR)	32	RW	0000_0000h
5_0800h	Block 3 status queue mode register (B3SQMR)	32	RW	0000_0000h
5_0804h	Block 3 status queue status register (B3SQSR)	32	RO	0000_0000h
5_0808h	Block 3 status queue extended dequeue pointer address register (B3SQEDPAR)	32	RW	0000_0000h
5_080Ch	Block 3 status queue dequeue pointer address register (B3SQDPAR)	32	RW	0000_0000h

Table continues on the next page...

## Memory Map

Offset	Register	Width (In bits)	Access	Reset value
5_0810h	Block 3 status queue extended enqueue pointer address register (B3SQEEPAR)	32	RW	0000_0000h
5_0814h	Block 3 status queue enqueue pointer address register (B3SQEPAR)	32	RW	0000_0000h
5_0828h	Block 3 status queue interrupt coalescing register (B3SQICR)	32	RW	0000_0000h

### 31.2.1.2 DMA mode register (DMR)

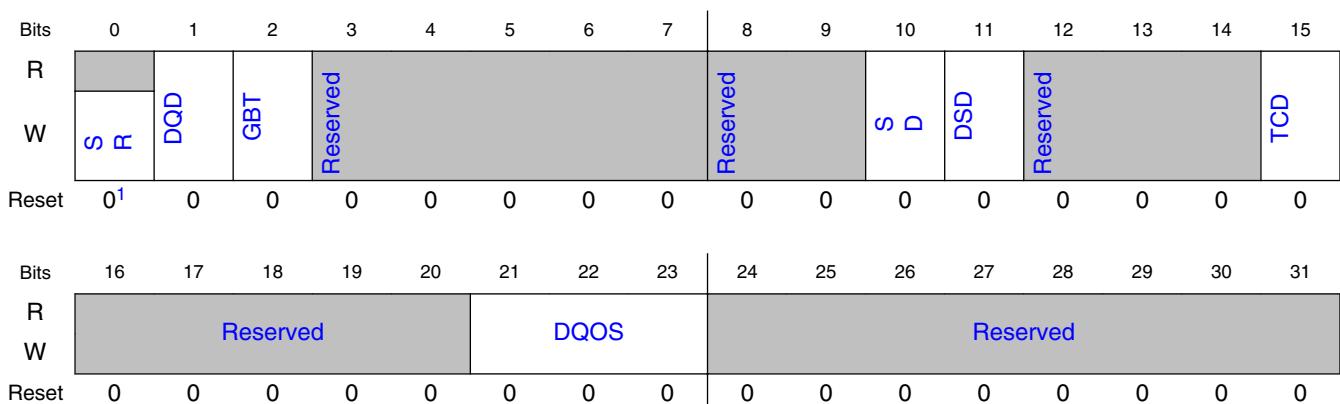
#### 31.2.1.2.1 Offset

Register	Offset
DMR	0h

#### 31.2.1.2.2 Function

The DMA mode register allows software to control global functions and capabilities of the DMA.

#### 31.2.1.2.3 Diagram



1. n

### 31.2.1.2.4 Fields

Field	Function
0 SR	Soft reset. Write only. Setting this bit causes reset of all registers and state for the qDMA controller, all state will be lost. Care should be taken when setting this bit as no outstanding requests must exist for proper operation after reset. Software should first try to reach a quiesce state by setting the DQD bit and wait for the qDMA to reach an idle state, DSR[DB]=0. If qDMA fails to reach an idle state, software should wait an appropriate amount of time for all responses to outstanding transactions to return before initiating a soft reset. Writing zero to this bit has no effect.
1 DQD	Dequeue disable. Valid only for command queue mode.  This bit can be used by software to bring qDMA into a quiesce state by polling the busy bit, DSR[DB]. 0b - Dequeue enable. qDMA controller will start processing the first DMA transaction automatically from command queues when available. 1b - Dequeue disable.
2 GBT	Global bandwidth throttle enable. Allows for control of qDMA bandwidth usage in the system using a token bucket allocation scheme. See . 0b - Disabled. 1b - Enabled.
3-7 —	Reserved
8-9 —	Reserved
10 SD	Snoop disable for memory data access. 0b - Snoop attribute depends on SD[SRTTTYPE] or DD[DWTTTYPE]. 1b - Snoop is disabled regardless of SD[SRTTTYPE] or DD[DWTTTYPE].
11 DSD	Snoop disable for memory descriptor access. 0b - Snoop is enabled. 1b - Snoop is disabled.
12-14 —	Reserved
15 TCD	Throttle Control Disable. 0b - Reading from or writing to slow memory interface are throttled by SD[RTHROTL] and DD[WTHROTL]. 1b - Throttle control is disabled, SD[RTHROTL] and DD[WTHROTL] are ignored.
16-20 —	Reserved
21-23 DQOS	QoS for memory descriptor access. qDMA does not use this for internal arbitration, it is only passed along with the transaction for possible quality of service at other arbitration/switch points to the target. 000 Lowest quality of service 001 Next lowest quality of service ... 110 Next highest quality of service 111 Highest quality of service
24-31 —	Reserved

### 31.2.1.3 DMA status register (DSR\_P)

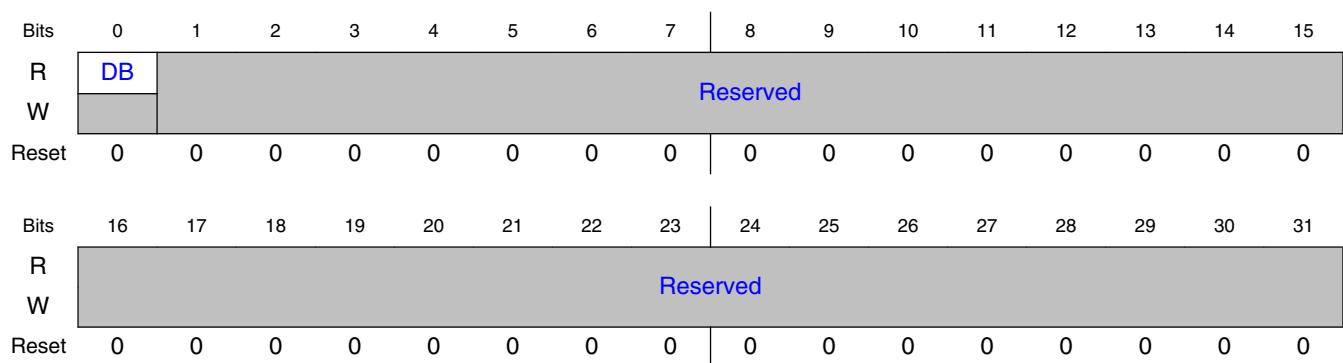
#### 31.2.1.3.1 Offset

Register	Offset
DSR_P	4h

#### 31.2.1.3.2 Function

The DMA status register reports the qDMA hardware engine status.

#### 31.2.1.3.3 Diagram



#### 31.2.1.3.4 Fields

Field	Function
0	DMA busy.
DB	0b - DMA idle. 1b - DMA busy.
1-31	Reserved
—	

### 31.2.1.4 DMA stream ID flush register (DSIDFR)

### 31.2.1.4.1 Offset

Register	Offset
DSIDFR	40h

### 31.2.1.4.2 Function

This is the DMA stream ID flush register which supports partition level reset assistance of pending jobs matching the stream ID given. This register is only applicable for command queue initiated transactions.

### 31.2.1.4.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	FM		Reserved													
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								STREAM_ID							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.4.4 Fields

Field	Function
0-1 FM	Flush mode. All other values reserved. 00b - Disabled. 01b - Stream ID match only. Pending and new DMA transactions matching stream ID will be terminated with no status notification.
2-23 —	Reserved
24-31 STREAM_ID	Stream ID.

### 31.2.1.5 DMA status register (DSR\_M)

## Memory Map

### 31.2.1.5.1 Offset

Register	Offset
DSR_M	1_0004h

### 31.2.1.5.2 Function

The DMA status register reports the qDMA hardware engine status.

### 31.2.1.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DB															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.5.4 Fields

Field	Function
0	DMA busy. 0b - DMA idle. 1b - DMA busy.
DB	Reserved
1-31 —	Reserved

## 31.2.1.6 DMA global bandwidth throttle register (DGBTR)

### 31.2.1.6.1 Offset

Register	Offset
DGBTR	1_0040h

### 31.2.1.6.2 Function

This is the DMA global bandwidth throttle register for controlling the qDMA global bandwidth usage in the system. The register defines the token bucket addition rate and token bucket capacity. For more information, see [Global Bandwidth Throttling](#).

The token bucket addition rate indicates how many bytes are added to the token bucket at a certain time interval. To calculate the token addition rate of bytes per second use:  $(2^{\text{MULT}} / 2^{\text{SCALE}}) / \text{platform\_frequency}$  bytes/s.

The token bucket capacity indicates the maximum number of tokens that can be accumulated. Any tokens accumulated above the capacity are dropped. One token is equal to one byte of DMA transfer. It is recommended that global bandwidth throttling be enabled after setting this register, for the token bucket counter to be initialized to the correct value.

### 31.2.1.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								TBC							
W	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				MULT				Reserved				SCALE			
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.6.4 Fields

Field	Function
0-11	Reserved
—	
12-15 TBC	Token bucket capacity (in bytes) defined by $2^{\text{TBC}}$ . Bit settings not shown are reserved. 1000b - 256 bytes 1001b - 512 bytes 1010b - 1K bytes 1011b - 2K bytes 1100b - 4K bytes
16-20	Reserved
—	
21-23	The token addition rate multiplier is $2^{\text{MULT}}$ .

Table continues on the next page...

## Memory Map

Field	Function
MULT	
24-28	Reserved
—	
29-31	The token addition rate scaler is $2^{\text{SCALE}}$ .
SCALE	

### 31.2.1.7 IP Block Revision register 0 (IPBRR0)

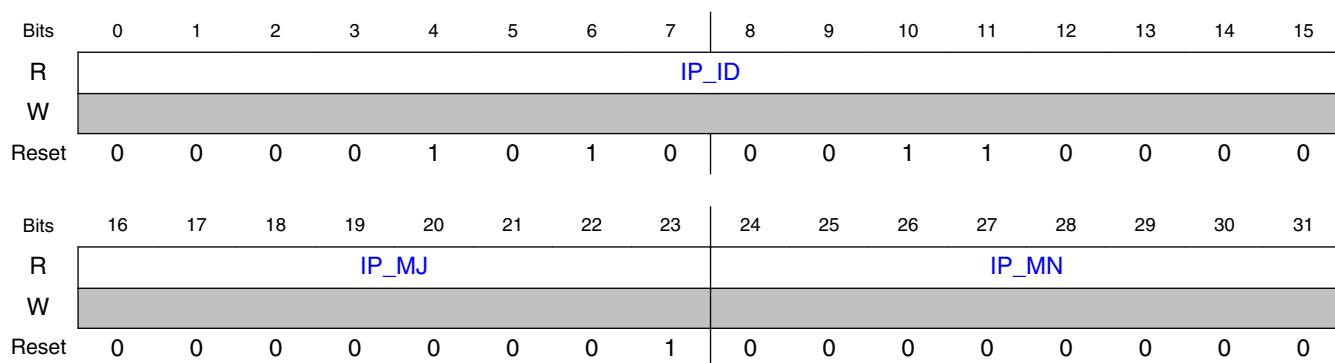
#### 31.2.1.7.1 Offset

Register	Offset
IPBRR0	1_0BF8h

#### 31.2.1.7.2 Function

The IP block revision register 0 (IPBRR0) is used to identify the qDMA block and revision.

#### 31.2.1.7.3 Diagram



#### 31.2.1.7.4 Fields

Field	Function
0-15 IP_ID	IP block ID

Table continues on the next page...

Field	Function
16-23 IP_MJ	Major revision
24-31 IP_MN	Minor revision

### 31.2.1.8 IP Block Revision register 1 (IPBRR1)

#### 31.2.1.8.1 Offset

Register	Offset
IPBRR1	1_0BFCh

#### 31.2.1.8.2 Function

The IP block revision register 1 (IPBRR1) is used to identify qDMA block configuration.

#### 31.2.1.8.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								IP_INT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	IP_MNT								IP_CFG							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 31.2.1.8.4 Fields

Field	Function
0-7 —	Reserved
8-15 IP_INT	IP block integration options

Table continues on the next page...

## Memory Map

Field	Function
16-23 IP_MNT	IP block maintenance version
24-31 IP_CFG	IP block configuration options. 7-6: DMA engines. 00 1 engine 01 2 engines 10 4 engines 11 8 engines 5-4: Reserved 3-0: Reserved

### 31.2.1.9 DMA error interrupt enable register (DEIER)

#### 31.2.1.9.1 Offset

Register	Offset
DEIER	1_0E00h

#### 31.2.1.9.2 Function

The DMA error interrupt enable register determines if a detected error condition should cause a system error interrupt. A system error interrupt occurs if a bit in this register is set and the corresponding bit in the error interrupt register is also set. To clear the interrupt, write a 1 to the error interrupt register.

### 31.2.1.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MEIE E	Reserved	RTEIE E	WTEIE E	CDEIE E	SDEIE E	DDEIE E	ERIEIE E	Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.9.4 Fields

Field	Function
0 MEIE	Multiple error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
1 —	Reserved
2 RTEIE	Read transaction error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
3 WTEIE	Write transaction error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
4 CDEIE	Command descriptor error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
5 SDEIE	Source descriptor error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
6 DDEIE	Destination descriptor error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
7 ERIEIE	Enqueue rejection error interrupt enable. 0b - Disabled. 1b - Interrupt enabled.
8-31 —	Reserved

### 31.2.1.10 DMA error detect register (DEDR)

#### 31.2.1.10.1 Offset

Register	Offset
DEDR	1_0E04h

#### 31.2.1.10.2 Function

The DMA error detect register indicates if an error condition was detected that should generate a system error interrupt. The error capture registers may hold additional information regarding the error. Write 1 to clear the error condition and the interrupt, if enabled.

#### 31.2.1.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ME	Reserved	RT E	WTE	CDE	SD E	DDE	ER E	Reserved							
W	W1C		W1C	W1C	W1C	W1C	W1C	W1C	Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 31.2.1.10.4 Fields

Field	Function
0 ME	Multiple errors of the same type.
1 —	Reserved
2 RTE	Read transaction error. A read memory access has caused a violation. The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively. Byte count successfully transferred is captured in DECBR.

*Table continues on the next page...*

Field	Function
3 WTE	Write transaction error. A write memory access has caused a violation. The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively. Byte count successfully transferred is captured in DECBR.
4 CDE	Command descriptor error. A format error was detected when parsing the frame descriptor. This includes: <ul style="list-style-type: none"> <li>• Invalid command descriptor format option.</li> </ul> The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively. Byte count successfully transferred is captured in DECBR.
5 SDE	Source descriptor error. This includes: <ul style="list-style-type: none"> <li>• Zero length</li> <li>• Illegal SD[SRTYPE]</li> <li>• S/G table early termination</li> </ul> The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively. Byte count successfully transferred is captured in DECBR.
6 DDE	Destination descriptor error. This includes: <ul style="list-style-type: none"> <li>• Illegal length (destination length exceeds source length)</li> <li>• Illegal DD[DWTTYPE]</li> <li>• S/G table early termination</li> </ul> The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively. Byte count successfully transferred is captured in DECBR.
7 ERE	Enqueue rejection error. An attempt to enqueue status notification result to status queue failed and the command descriptor was returned. Failure to handle this error may result in delay of subsequent enqueue operations which could lead to a halt of DMA operations. The command descriptor and command queue ID associated with the transaction has been captured in DECCDR0-3 and DECCQIDR respectively.
8-31 —	Reserved

### 31.2.1.11 DMA error capture frame descriptor word 0 register (DECC DW0R)

#### 31.2.1.11.1 Offset

Register	Offset
DECCDW0R	1_0E10h

#### 31.2.1.11.2 Function

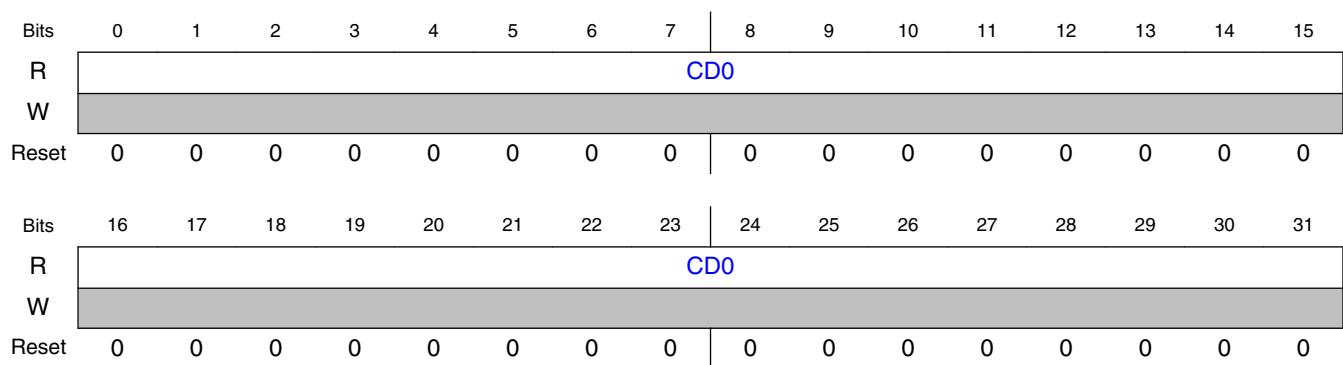
This is the DMA error capture command descriptor registers. Software must clear all error conditions listed to capture the next command descriptor. The CD[STATUS] bits are not updated to reflect the error condition.

## Memory Map

The following error conditions will capture the frame descriptor:

- Read transaction error - RTE
- Write transaction error - WTE
- Command descriptor error - CDE
- Source descriptor error - SDE
- Destination descriptor error - DDE
- Enqueue rejection error - ERE

### 31.2.1.11.3 Diagram



### 31.2.1.11.4 Fields

Field	Function
0-31	Command descriptor word 0 associated with the error (bits 0-31).
CD0	

## 31.2.1.12 DMA error capture frame descriptor word 1 register (DECC DW1R)

### 31.2.1.12.1 Offset

Register	Offset
DECCDW1R	1_0E14h

### 31.2.1.12.2 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									CD1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CD1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.12.3 Fields

Field	Function
0-31	Command descriptor word 1 associated with the error (bits 32-63).
CD1	

## 31.2.1.13 DMA error capture frame descriptor word 2 register (DECC DW2R)

### 31.2.1.13.1 Offset

Register	Offset
DECCDW2R	1_0E18h

### 31.2.1.13.2 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									CD2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									CD2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.13.3 Fields

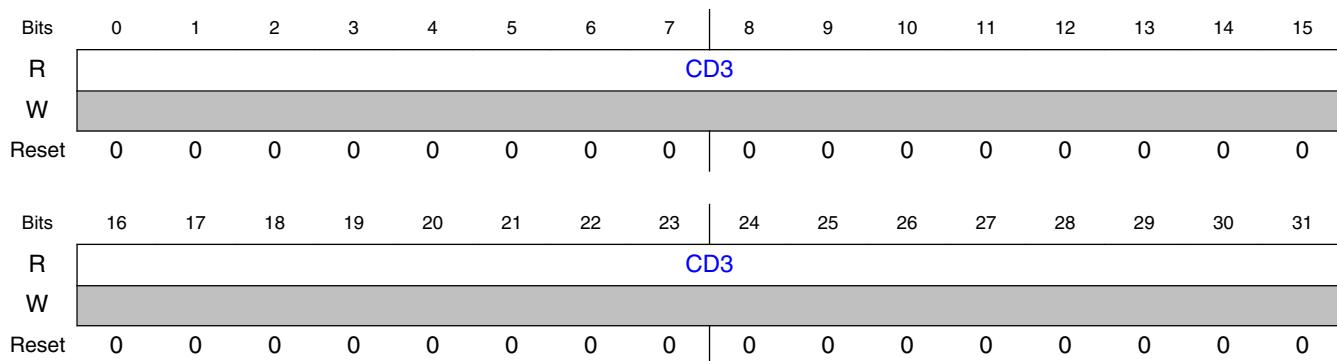
Field	Function
0-31	Command descriptor word 2 associated with the error (bits 64-95).
CD2	

### 31.2.1.14 DMA error capture frame descriptor word 3 register (DECC DW3R)

#### 31.2.1.14.1 Offset

Register	Offset
DECCDW3R	1_0E1Ch

#### 31.2.1.14.2 Diagram



#### 31.2.1.14.3 Fields

Field	Function
0-31	Command descriptor word 3 associated with the error (bits 96-127).
CD3	

### 31.2.1.15 DMA error capture command queue ID register (DECC QIDR)

#### 31.2.1.15.1 Offset

Register	Offset
DECCQIDR	1_0E30h

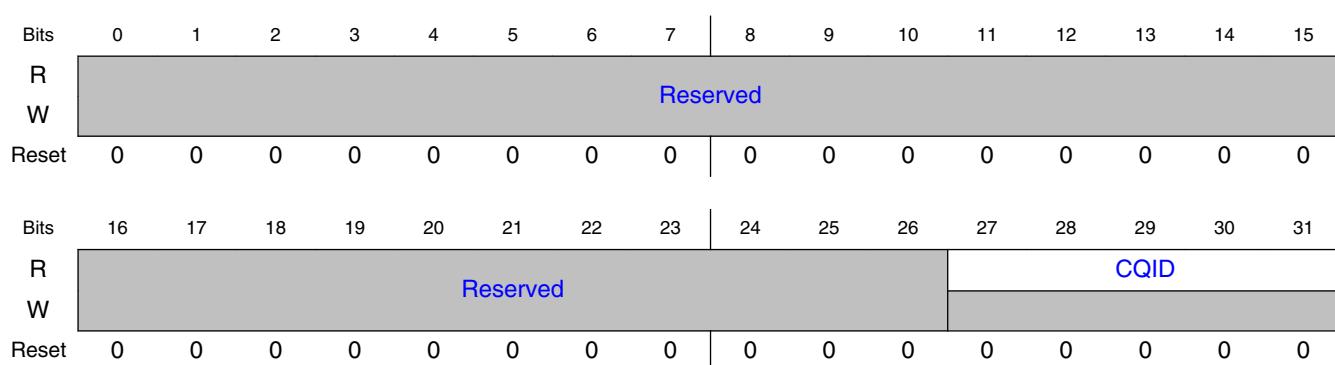
#### 31.2.1.15.2 Function

This is the DMA error capture command queue ID register. Reading this register will determine the command queue ID associated with the error frame descriptor captured. This value represent the producer of the DMA transfer. Software must clear all error conditions listed to capture the next command queue ID.

The following error conditions will capture the command queue ID:

- Read transaction error - RTE
- Write transaction error - WTE
- Command descriptor error - CDE
- Source descriptor error - SDE
- Destination descriptor error - DDE
- Enqueue rejection error - ERE

#### 31.2.1.15.3 Diagram



#### 31.2.1.15.4 Fields

Field	Function
0-26	Reserved

*Table continues on the next page...*

## Memory Map

Field	Function
—	
27-31	Command queue ID. Read only.
CQID	Bit
26-27	Block 0-3
29-31	Queue number 0-7

## 31.2.1.16 DMA error capture byte count register (DECBR)

### 31.2.1.16.1 Offset

Register	Offset
DECBR	1_0E34h

### 31.2.1.16.2 Function

This is the DMA error capture byte count register. This register is used to record the number of bytes that has been transferred prior to error detected. Note that the byte count is the exact byte count written to destination. Software must clear all error conditions listed to capture the next byte count.

The following error conditions will capture the byte count:

- Read transaction error - RTE
- Write transaction error - WTE
- Command descriptor error - CDE
- Source descriptor error - SDE
- Destination descriptor error - DDE

### 31.2.1.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									BC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									BC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.16.4 Fields

Field	Function
0-31	This is used to record number of bytes that have been transferred prior to the error.
BC	

## 31.2.1.17 Block a command queue b mode register (B0CQ0MR - B3CQ7MR)

### 31.2.1.17.1 Offset

For a = 0 to 3; b = 0 to 7:

Register	Offset
BaCQbMR	2_00C0h + (a × 1_0000h) + (b × 100h)

### 31.2.1.17.2 Function

The command queue mode register allows software to control various command queue operation characteristics.

### 31.2.1.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EN	EI	Reserved					CD_THLD				CQ_SIZE				
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.17.4 Fields

Field	Function
0 EN	Command queue enable. 0b - Disabled. 1b - Enabled. Command queue must be initialized. When the queue is not empty, BaCQbSR[QE]=0 and no errors are pending, CQEDR / BaCQbIDR, transactions will be processed from the queue.
1 EI	Enqueue pointer increment. Setting this bit will cause the enqueue pointer BaCQbEPAR to increment to the next entry. Always reads as 0.
2-7 —	Reserved
8-11 CD_THLD	Interrupt threshold. Determines the threshold for the number of unprocessed command descriptors remaining before Command Queue Threshold interrupt is signaled. Undefined operation results if the message queue threshold is set greater than or equal to the message queue size (BaCQbMR[CQ_SIZE]).  For proper operation, this field should only be modified when the command queue is not enabled.  Bit settings not shown are reserved.  0000b - 16 0001b - 32 0010b - 64 0011b - 128 0100b - 256 0101b - 512 0110b - 1024 0111b - 2048 1000b - 4096 1001b - 8192
12-15 CQ_SIZE	Circular descriptor queue size. Determines the number of command descriptors that can be placed on the circular queue without overflow.  For proper operation, this field should only be modified when the command queue is not enabled  Bit settings not shown here are reserved.  0000b - 64 0001b - 128

Table continues on the next page...

Field	Function
	0010b - 256 0011b - 512 0100b - 1024 0101b - 2048 0110b - 4096 0111b - 8192 1000b - 16384
16-31 —	Reserved

### 31.2.1.18 Block a command queue b status register (B0CQ0SR - B3CQ7SR)

#### 31.2.1.18.1 Offset

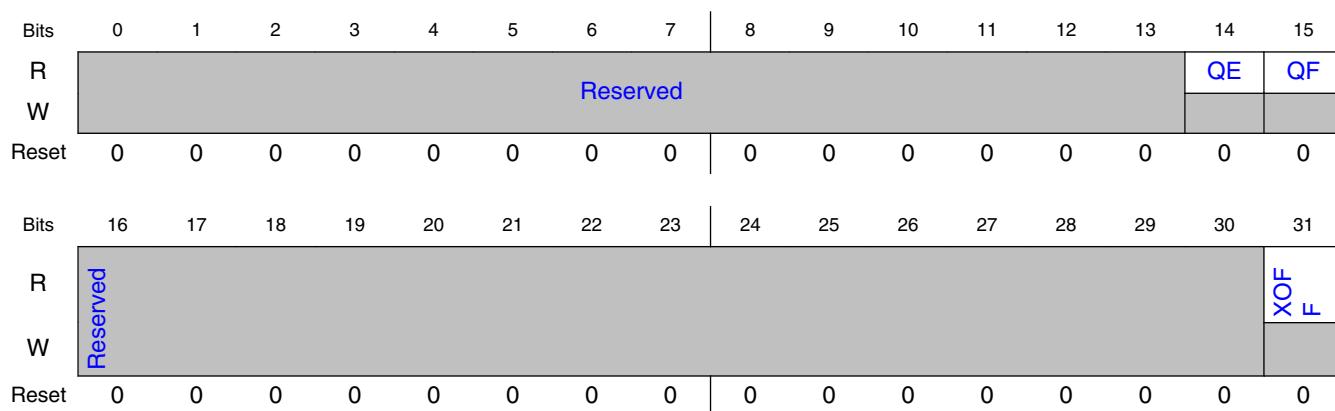
For a = 0 to 3; b = 0 to 7:

Register	Offset
BaCQbSR	2_00C4h + (a × 1_0000h) + (b × 100h)

#### 31.2.1.18.2 Function

The command queue status register reports various queue conditions during and after enqueue/dequeue operations. Valid only when queue is enabled.

#### 31.2.1.18.3 Diagram



### 31.2.1.18.4 Fields

Field	Function
0-13 —	Reserved
14 QE	Queue empty. Read-only. This bit is automatically cleared if the command queue is disabled. 0b - Queue is not empty. 1b - Queue is empty.
15 QF	Queue full. Read-only. This bit is automatically cleared if the command queue is disabled. 0b - Queue is not full. 1b - Queue is full.
16-30 —	Reserved
31 XOFF	Command queue has entered congestion management (XOFF), temporarily halting further processing of transactions. The XOFF state is an indication status queue has reached set critical level. To exit congestion state, the status queue must be below critical level. This bit is automatically cleared if the command queue is disabled.

### 31.2.1.19 Block a command queue b extended dequeue pointer address register (B0CQ0EDPAR - B3CQ7EDPAR)

#### 31.2.1.19.1 Offset

For a = 0 to 3; b = 0 to 7:

Register	Offset
BaCQbEDPAR	2_00C8h + (a × 1_0000h) + (b × 100h)

#### 31.2.1.19.2 Function

The command queue dequeue pointer address registers contain the address of the first command descriptor in memory to be processed. Software must initialize these registers to point to the first command descriptor in memory. After processing this descriptor, the command queue increments the dequeue pointer address in BaCQb(E)DPAR to point to the next descriptor. If the command queue enqueue pointer and the command queue dequeue pointer are not equal (indicating that the queue is not empty), the command queue reads the next descriptor from memory for processing. If the enqueue and dequeue pointers are equal after the dequeue pointer has been incremented by the message queue,

the queue is empty and further processing halts until the enqueue pointer is incremented by the processor. Incrementing the pointer indicates that a new descriptor has been added to the queue and is ready for transmission. If the queue becomes empty, BaCQbSR[QE]=1 is set. If the read of the command descriptor causes an internal transaction error, the transaction error condition will occur and the pointer will not be updated. If CQIER[TEIE]=1 and CQEDR[TE]=1, a transaction error interrupt will be generated.

When software initializes the dequeue pointer, the queue to which it points must be aligned on a boundary equal to the number of queue entries x command descriptor size. For example, if there are 64 entries, the queue must be 1024-byte aligned.

### 31.2.1.19.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								ECQDPA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.19.4 Fields

Field	Function
0-23	Reserved
24-31 ECQDPA	Extended command queue dequeue pointer address bits. These are the highest order address bits. For proper operation, this field should only be modified when the command queue is not enabled.

## 31.2.1.20 Block a command queue b dequeue pointer address register (B0CQ0DPAR - B3CQ7DPAR)

### 31.2.1.20.1 Offset

For a = 0 to 3; b = 0 to 7:

## Memory Map

Register	Offset
BaCQbDPAR	2_00CCh + (a × 1_0000h) + (b × 100h)

### 31.2.1.20.2 Diagram



### 31.2.1.20.3 Fields

Field	Function
0-27 CQDPA	Command queue dequeue pointer address. Contains the address of the first command descriptor in memory to process. The descriptor must be aligned to a 16 byte boundary. For proper operation, this field should only be modified when the command queue is not enabled.
28-31 —	Reserved

## 31.2.1.21 Block a command queue b extended enqueue pointer address register (B0CQ0EEPAR - B3CQ7EEPAR)

### 31.2.1.21.1 Offset

For a = 0 to 3; b = 0 to 7:

Register	Offset
BaCQbEEPAR	2_00D0h + (a × 1_0000h) + (b × 100h)

### 31.2.1.21.2 Function

The command queue enqueue pointer address registers contain the address for the next command descriptor in memory to be added to the queue. Software must initialize these registers to match the command queue dequeue pointer address. When a command is ready to be sent, the processor writes a command descriptor to the next location in the queue (indicated by the address in BaCQb(E)EPAR), and then either writes BaCQb(E)EPAR to point to the next descriptor location in memory or sets BaCQbMR[EI]. The queue is full when the enqueue pointer and the dequeue pointer are equal. Writing the enqueue pointer to a value outside of the queue size boundary will result in undefined behavior.

When software initializes the enqueue pointer, the queue to which it points must be aligned on a boundary equal to the number of queue entries x command descriptor size. For example, if there are 64 entries, the queue must be 1024-byte aligned.

### 31.2.1.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								ECQEPA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.21.4 Fields

Field	Function
0-23	Reserved
—	
24-31	Extended command queue enqueue pointer address. These are the highest order address bits.
ECQEPA	

## 31.2.1.22 Block a command queue b enqueue pointer address register (B0CQ0EPAR - B3CQ7EPAR)

## Memory Map

### 31.2.1.22.1 Offset

For a = 0 to 3; b = 0 to 7:

Register	Offset
BaCQbEPAR	2_00D4h + (a × 1_0000h) + (b × 100h)

### 31.2.1.22.2 Diagram



### 31.2.1.22.3 Fields

Field	Function
0-27 CQEPA	Command queue enqueue pointer address. Contains the address of the last command descriptor in memory to process. The descriptor must be aligned to a 16 byte boundary and a command queue boundary.
28-31 —	Reserved

## 31.2.1.23 Block a command queue interrupt enable registers (B0CQIER - B3CQIER)

### 31.2.1.23.1 Offset

Register	Offset
B0CQIER	2_00E0h
B1CQIER	3_00E0h
B2CQIER	4_00E0h
B3CQIER	5_00E0h

### 31.2.1.23.2 Function

The command queue interrupt enable register determines if an event should cause an interrupt. An interrupt occurs if a bit in this register is set and the corresponding bit in the interrupt detect register is also set.

To clear an interrupt, write a 1 to the interrupt detect register. Interrupts caused by normal status conditions are cleared by responding to the condition for which the interrupt was caused.

### 31.2.1.23.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CQ0TIE	CQ1TIE	CQ2TIE	CQ3TIE	CQ4TIE	CQ5TIE	CQ6TIE	CQ7TIE	SQPEI E	Reserved						
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SQTI E	Reserved							0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.23.4 Fields

Field	Function
0 CQ0TIE	Command queue 0 threshold interrupt enable. 0b - Disabled 1b - Command queue 0 threshold interrupt enabled.
1 CQ1TIE	Same as CQ0TIE.
2 CQ2TIE	Same as CQ0TIE.
3 CQ3TIE	Same as CQ0TIE.
4 CQ4TIE	Same as CQ0TIE.
5	Same as CQ0TIE.

Table continues on the next page...

## Memory Map

Field	Function
CQ5TIE	
6	Same as CQ0TIE.
CQ6TIE	
7	Same as CQ0TIE.
CQ7TIE	
8	Status queue programming error interrupt enable. 0b - Disabled 1b - Status queue programming error interrupt enabled.
9-15	Reserved
—	
16	Status queue threshold interrupt enable. 0b - Disabled 1b - Status queue threshold interrupt enabled.
SQTIE	
17-31	Reserved
—	

### 31.2.1.24 Block a command queue interrupt detect registers (B0CQ IDR - B3CQIDR)

#### 31.2.1.24.1 Offset

Register	Offset
B0CQIDR	2_00E4h
B1CQIDR	3_00E4h
B2CQIDR	4_00E4h
B3CQIDR	5_00E4h

#### 31.2.1.24.2 Function

The command queue interrupt detect register indicates if an event was detected. Write 1 to clear the detected event and the interrupt, if enabled.

### 31.2.1.24.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CQ0T	CQ1T	CQ2T	CQ3T	CQ4T	CQ5T	CQ6T	CQ7T	SQP	E	Reserved					
W	W1C	W1C	W1C	Reserved												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	SQT	Reserved														
W	W1C	Reserved														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.24.4 Fields

Field	Function
0 CQ0T	Command queue 0 threshold. Write 1 to clear. 0b - Command queue 0 threshold has not been crossed. 1b - Command queue 0 threshold has been crossed and less than BaCQ0MR[CD_THLD] entries remains in the queue.
1 CQ1T	Same as CQ0T.
2 CQ2T	Same as CQ0T.
3 CQ3T	Same as CQ0T.
4 CQ4T	Same as CQ0T.
5 CQ5T	Same as CQ0T.
6 CQ6T	Same as CQ0T.
7 CQ7T	Same as CQ0T.
8 SQPE	Status queue programming error. Status queue was disabled during an enqueue operation. Write 1 to clear.
9-15 —	Reserved
16 SQT	Status notification queue threshold. Write 1 to clear. 0b - Status notification queue threshold not reached.

Table continues on the next page...

## Memory Map

Field	Function
	1b - Status notification queue holds at least the number of commands specified by BaCQICR[ICMT] or the threshold timer has expired as specified by BaCQICR[ICTT].
17-31 —	Reserved

## 31.2.1.25 Block a status queue mode register (B0SQMR - B3SQMR)

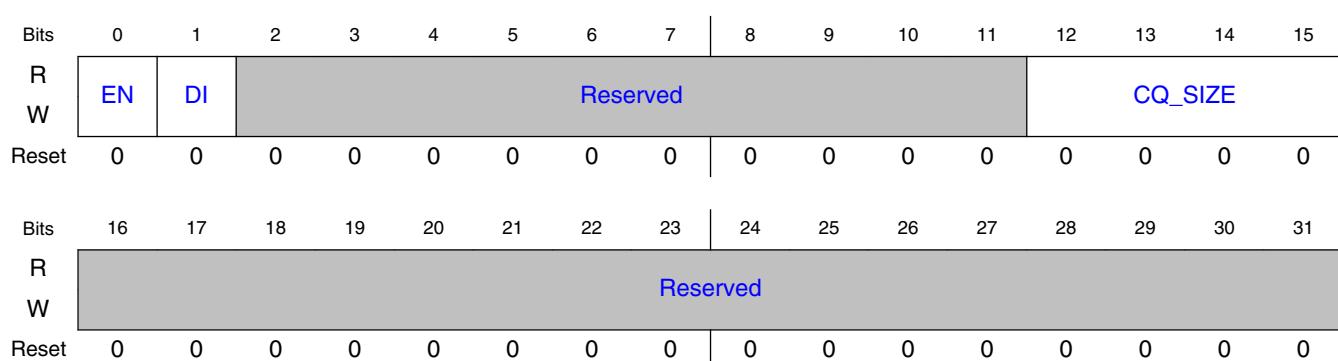
### 31.2.1.25.1 Offset

Register	Offset
B0SQMR	2_0800h
B1SQMR	3_0800h
B2SQMR	4_0800h
B3SQMR	5_0800h

### 31.2.1.25.2 Function

The status queue mode register allows software to control various status queue operation characteristics.

### 31.2.1.25.3 Diagram



### 31.2.1.25.4 Fields

Field	Function
0	Status queue enable.

*Table continues on the next page...*

Field	Function
EN	For proper operation, this bit should not be cleared while command queues are steering traffic to this status queue. 0b - Disabled. 1b - Enabled. Status queue has been initialized and can service command queues.
1 DI	Dequeue pointer increment. Setting this bit will cause the dequeue pointer BaSQDPAR to increment to the next entry. Always reads as 0.
2-11 —	Reserved
12-15 CQ_SIZE	Circular descriptor queue size. Determines the number of command descriptors that can be placed on the circular queue without overflow. For proper operation, this field should only be modified when the status queue is not enabled. 0000 64 0001 128 0010 256 0011 512 0100 1024 0101 2048 0110 4096 0111 8192 1000 16384 1001 Reserved ... 1111 Reserved
16-31 —	Reserved

### 31.2.1.26 Block a status queue status register (B0SQSR - B3SQSR)

#### 31.2.1.26.1 Offset

Register	Offset
B0SQSR	2_0804h
B1SQSR	3_0804h
B2SQSR	4_0804h
B3SQSR	5_0804h

## Memory Map

### 31.2.1.26.2 Function

The status queue status register reports various queue conditions during and after enqueue/dequeue operations. Valid only when queue is enabled.

### 31.2.1.26.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R															QE	QF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.26.4 Fields

Field	Function
0-13	Reserved
—	
14	Queue empty. Read-only.
QE	0b - Queue is not empty. 1b - Queue is empty.
15	Queue full. Read-only.
QF	0b - Queue is not full. 1b - Queue is full.
16-31	Reserved
—	

### 31.2.1.27 Block a status queue extended dequeue pointer address register (B0SQEDPAR - B3SQEDPAR)

#### 31.2.1.27.1 Offset

Register	Offset
B0SQEDPAR	2_0808h

Table continues on the next page...

Register	Offset
B1SQEDPAR	3_0808h
B2SQEDPAR	4_0808h
B3SQEDPAR	5_0808h

### 31.2.1.27.2 Function

The status queue dequeue pointer address registers contain the address of the first command descriptor in memory to be processed by software. Software must initialize these registers to point to the first command descriptor in memory. After processing this command descriptor, software either write BaSQ(E)DPAR to point to the next descriptor location in memory or sets BaSQMR[DI] to cause QMLite to increment BaSQ(E)DPAR to point to the next descriptor. The status queue is not empty if the enqueue pointers and the dequeue pointers are not equal. This is also reflected by the status bit BaSQSR[QE] being clear. If the two pointers are equal, it could mean the queue is either full or empty and the status register BaSQSR should be read to avoid an underflow. Writing the dequeue pointer to a value outside of the queue size boundary will results in undefined behavior.

When software initializes the dequeue pointer, the queue to which it points must be aligned on a boundary equal to the number of queue entries x command descriptor size. For example, if there are 64 entries, the queue must be 1024-byte aligned.

### 31.2.1.27.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								SQDPA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.27.4 Fields

Field	Function
0-23	Reserved

Table continues on the next page...

## Memory Map

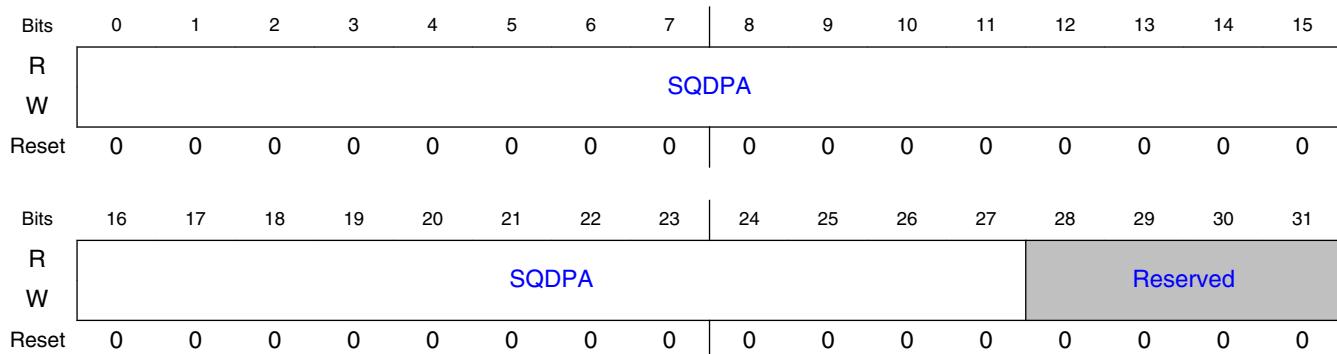
Field	Function
—	
24-31 SQDPA	Upper status queue dequeue pointer address bits.

## 31.2.1.28 Block a status queue dequeue pointer address register (B0SQDPAR - B3SQDPAR)

### 31.2.1.28.1 Offset

Register	Offset
B0SQDPAR	2_080Ch
B1SQDPAR	3_080Ch
B2SQDPAR	4_080Ch
B3SQDPAR	5_080Ch

### 31.2.1.28.2 Diagram



### 31.2.1.28.3 Fields

Field	Function
0-27 SQDPA	Status queue dequeue pointer address. Contains the address of the first command descriptor in memory to process. The command descriptor must be aligned to a 16 byte boundary.
28-31 —	Reserved

### 31.2.1.29 Block a status queue extended enqueue pointer address register (B0SQEEPAR - B3SQEEPAR)

#### 31.2.1.29.1 Offset

Register	Offset
B0SQEEPAR	2_0810h
B1SQEEPAR	3_0810h
B2SQEEPAR	4_0810h
B3SQEEPAR	5_0810h

#### 31.2.1.29.2 Function

The status queue enqueue pointer address registers contain the address of the next command descriptor to be added by the status queue. Software must initialize these registers to point to the first command descriptor in memory. After adding a new command descriptor, the status queue increments the enqueue pointer address in BaSQ(E)EPAR to point to the next descriptor. The status queue is not empty if the enqueue pointers and the dequeue pointers are not equal. This is also reflected by the status bit BaSQSR[QE] being clear. If the two pointers are equal, it could mean the queue is either full or empty and the status register BaSQSR should be read.

If the queue becomes full when QMLite writes a new command descriptor, the queue full status condition will occur, BaSQSR[QF]=1. An internal critical level watermark will stop the completion queue from reaching overflow condition. If this watermark has been reached, outbound dequeue is automatically halted for all queues associated with the status queue. Software should enable the status queue threshold detect in advance to avoid reaching this condition. If the write of the status descriptor causes an internal transaction error, the transaction error condition will occur. If CQIER[TEIE]=1 and CQEDR[TE]=1, a transaction error interrupt will be generated.

When software initializes the enqueue pointer, the queue to which it points must be aligned on a boundary equal to the number of queue entries x command descriptor size. For example, if there are 64 entries, the queue must be 1024-byte aligned.

## Memory Map

### 31.2.1.29.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved					SQEPA							
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.29.4 Fields

Field	Function
0-23	Reserved
—	
24-31	Extended status queue enqueue pointer address bits. These are the highest order address bits.
SQEPA	

### 31.2.1.30 Block a status queue enqueue pointer address register (B0SQEPAR - B3SQEPAR)

#### 31.2.1.30.1 Offset

Register	Offset
B0SQEPAR	2_0814h
B1SQEPAR	3_0814h
B2SQEPAR	4_0814h
B3SQEPAR	5_0814h

### 31.2.1.30.2 Diagram



### 31.2.1.30.3 Fields

Field	Function
0-27 SQEPA	Status queue enqueue pointer address. Contains the address of the next command descriptor to be added by QMLite.
28-31 —	Reserved

## 31.2.1.31 Block a status queue interrupt coalescing register (B0SQICR - B3SQICR)

### 31.2.1.31.1 Offset

Register	Offset
B0SQICR	2_0828h
B1SQICR	3_0828h
B2SQICR	4_0828h
B3SQICR	5_0828h

### 31.2.1.31.2 Function

The status queue interrupt coalescing register enables and configures the parameters for interrupt coalescing associated with received commands.

## Memory Map

### 31.2.1.31.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	ICEN	Reserved								ICST							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	ICTT																
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 31.2.1.31.4 Fields

Field	Function
0 ICEN	Interrupt coalescing enable. 0b - Interrupt coalescing is disabled. 1b - Interrupt coalescing is enabled. If the status queue threshold interrupt is enabled (BaCQIER[SQTIE] is set), an interrupt is raised when the threshold number of status notifications is reached (defined by BaSQICR[ICMT]) or when the threshold timer expires (determined by BaSQICR[ICTT]).
1-11 —	Reserved
12-15 ICST	Interrupt coalescing status threshold. While interrupt coalescing is enabled (BaSQICR[ICEN] is set), this value determines the minimum number of transactions present in the status queue before raising an interrupt. Software should set the dequeue pointer after processing the event such that the number of transactions in the queue is less than the threshold to avoid further interrupts.  For proper operation, this field should only be modified when the status queue is not enabled.  0000 0 (disabled) 0001 1 status notification 0010 2 status notifications 0011 4 status notifications 0100 8 status notifications 0101 16 status notifications 0110 32 status notifications 0111 64 status notifications 1000 128 status notifications 1001 256 status notifications 1010 512 status notifications 1011 1024 status notifications 1100 2048 status notifications

Table continues on the next page...

Field	Function
	1101 Reserved ... 1111 Reserved
16-31 ICTT	Interrupt coalescing timer threshold. While interrupt coalescing is enabled (BaSQICR[ICEN] is set), this value determines the maximum amount of time after receiving a status notification before raising an interrupt. If status notifications have been received but the status notification threshold has not been met, an interrupt is raised when the threshold timer reaches zero. The threshold timer is reset to this value upon receiving a status notification.

### 31.2.1.32 Command queue mode register (CQMR)

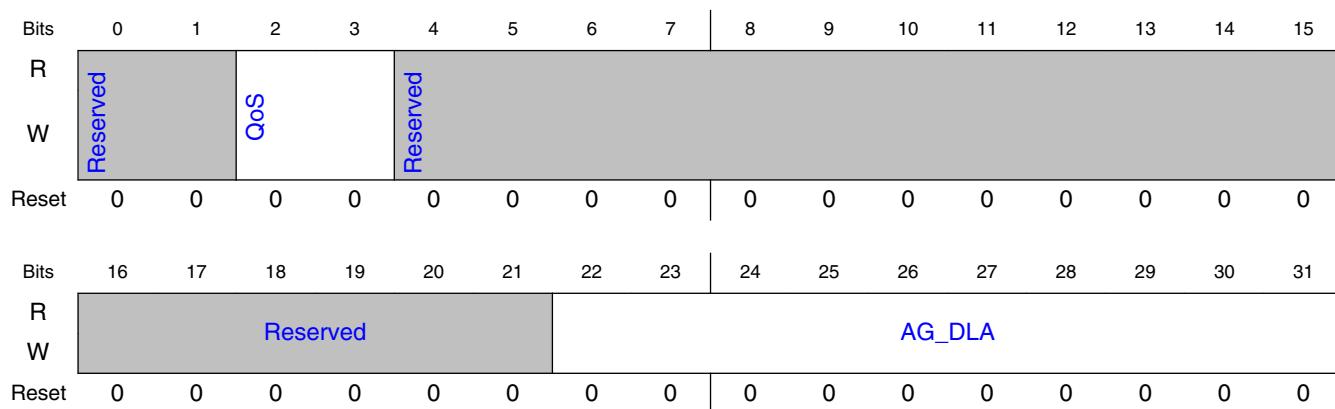
#### 31.2.1.32.1 Offset

Register	Offset
CQMR	2_0A00h

#### 31.2.1.32.2 Function

The mode register allows for global control of system access parameters.

#### 31.2.1.32.3 Diagram



#### 31.2.1.32.4 Fields

Field	Function
0-1	Reserved

Table continues on the next page...

## Memory Map

Field	Function
—	
2-3 QoS	QoS used for memory access (0-7). A higher value indicates better quality of service. 00b - 0 01b - 2 10b - 4 11b - 6
4-21	Reserved
—	
22-31 AG_DLA	Arbitration group deadlock avoidance. This field controls the deadlock avoidance threshold for arbitration group selection. Deadlock avoidance is applied if a lower priority request has lost arbitration N number of times. A request that has reached the deadlock watermark will Round-Robin with other request in the same state and has priority over normal requests. A zero value will disable the deadlock avoidance mechanism and perform strict priority only.

### 31.2.1.33 Command queue dequeue scheduler configuration register 1 (CQDSCR1)

#### 31.2.1.33.1 Offset

Register	Offset
CQDSCR1	2_0A08h

#### 31.2.1.33.2 Function

The dequeue scheduler configuration register 1 is used to configure the arbitration group of command queues based on the arbitration scheme described in [Dequeue Scheduling](#). The maximum number of arbitration groups used in the system is defined by the highest assigned arbitration group number.

### 31.2.1.33.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	AGQ0		Reserved	AGQ1		Reserved	AGQ2		Reserved	AGQ3					
W	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	AGQ4		Reserved	AGQ5		Reserved	AGQ6		Reserved	AGQ7					
W	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1
Reset	0	1	0	0	0	1	0	1	0	1	1	0	0	1	1	1

### 31.2.1.33.4 Fields

Field	Function
0 —	Reserved
1-3 AGQ0	Arbitration group number for command queue 0, see <a href="#">Dequeue Scheduling</a> . This field must be zero
4 —	Reserved
5-7 AGQ1	Arbitration group number for command queue 1. Same or one higher than AGQ0.
8 —	Reserved
9-11 AGQ2	Arbitration group number for command queue 2. Same or one higher than AGQ1.
12 —	Reserved
13-15 AGQ3	Arbitration group number for command queue 3. Same or one higher than AGQ2.
16 —	Reserved
17-19 AGQ4	Arbitration group number for command queue 4. Same or one higher than AGQ3.
20 —	Reserved
21-23	Arbitration group number for command queue 5. Same or one higher than AGQ4.

Table continues on the next page...

## Memory Map

Field	Function
AGQ5	
24 —	Reserved
25-27	Arbitration group number for command queue 6. Same or one higher than AGQ5.
AGQ6	
28 —	Reserved
29-31	Arbitration group number for command queue 7. Same or one higher than AGQ6.
AGQ7	This field determines the highest numbered arbitration group in the system.

### 31.2.1.34 Command queue dequeue scheduler configuration register 2 (CQDSCR2)

#### 31.2.1.34.1 Offset

Register	Offset
CQDSCR2	2_0A0Ch

#### 31.2.1.34.2 Function

The dequeue scheduler configuration register 2 is used to configure the weight selection of command queues based on the arbitration scheme described in [Dequeue Scheduling](#).

#### 31.2.1.34.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	AWQ0			Reserved	AWQ1			Reserved	AWQ2			Reserved	AWQ3		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	AWQ4			Reserved	AWQ5			Reserved	AWQ6			Reserved	AWQ7		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.34.4 Fields

Field	Function
0 —	Reserved
1-3 AWQ0	Arbitration weight for command queue 0. Assigned weight = AWQ0, range 0 to 7. See <a href="#">Dequeue Scheduling</a> .
4 —	Reserved
5-7 AWQ1	Arbitration weight for command queue 1.
8 —	Reserved
9-11 AWQ2	Arbitration weight for command queue 2.
12 —	Reserved
13-15 AWQ3	Arbitration weight for command queue 3.
16 —	Reserved
17-19 AWQ4	Arbitration weight for command queue 4.
20 —	Reserved
21-23 AWQ5	Arbitration weight for command queue 5.
24 —	Reserved
25-27 AWQ6	Arbitration weight for command queue 6.
28 —	Reserved
29-31 AWQ7	Arbitration weight for command queue 7.

### 31.2.1.35 Command queue interrupt enable register (CQIER)

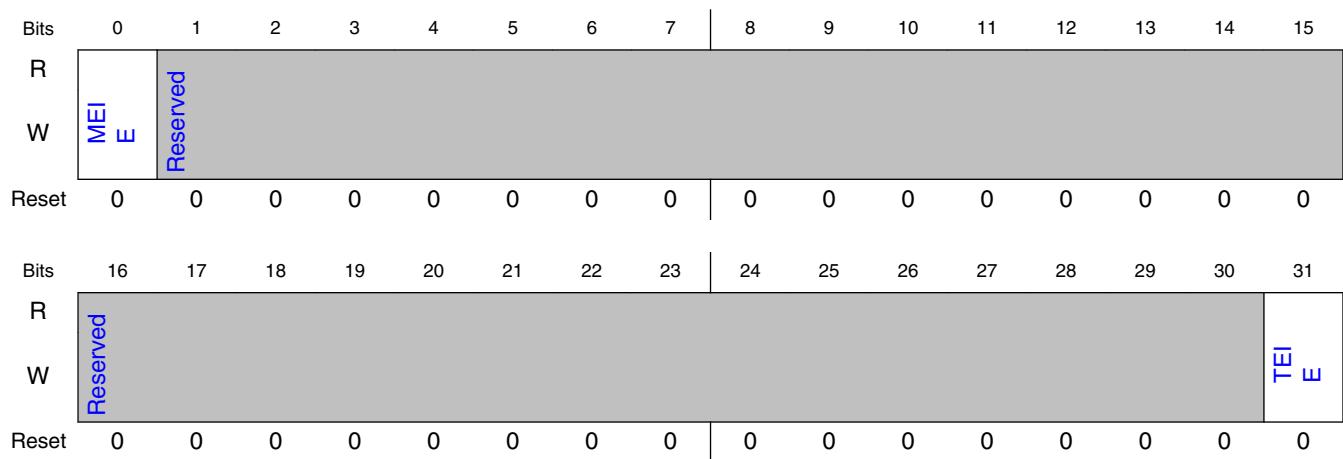
### 31.2.1.35.1 Offset

Register	Offset
CQIER	2_0A10h

### 31.2.1.35.2 Function

The command queue interrupt enable register determines if a detected error condition should cause an interrupt. An interrupt occurs if a bit in this register is set and the corresponding bit in the error detect register is also set. To clear the interrupt, write a 1 to the error detect register.

### 31.2.1.35.3 Diagram



### 31.2.1.35.4 Fields

Field	Function
0	Multiple error interrupt enable. 0b - Disabled. 1b - Multiple error interrupt enabled.
MEIE	Reserved
1-30	—
31	Transaction error interrupt enable. 0b - Disabled. 1b - Transaction error interrupt enabled.
TEIE	

### 31.2.1.36 Command queue error detect register (CQEDR)

#### 31.2.1.36.1 Offset

Register	Offset
CQEDR	2_0A14h

#### 31.2.1.36.2 Function

The command queue error detect register indicates if an error condition was detected. Detection of a condition can be disabled through the error capture disable register. Some errors will cause the error capture registers to be updated with additional information. Write 1 to clear the detected condition and the interrupt, if enabled.

#### 31.2.1.36.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ME	Reserved														
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved														TE	
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 31.2.1.36.4 Fields

Field	Function
0 ME	Multiple error. Multiple errors of the same kind reported. Write 1 to clear. The following conditions will set this bit: <ul style="list-style-type: none"> <li>Transaction error encountered while previously set.</li> </ul>
1-30 —	Reserved
31 TE	Transaction error. A read/write access to memory has caused an access violation. The Command Queue Error Capture Address Registers (CQECAr) has captured the memory address. Write 1 to clear. The queue responsible, as determined by the address, should be disabled by software before clearing the TE error to avoid additional errors occurring.

### 31.2.1.37 Command queue error capture extended address register (CQECEAR)

#### 31.2.1.37.1 Offset

Register	Offset
CQECEAR	2_0A18h

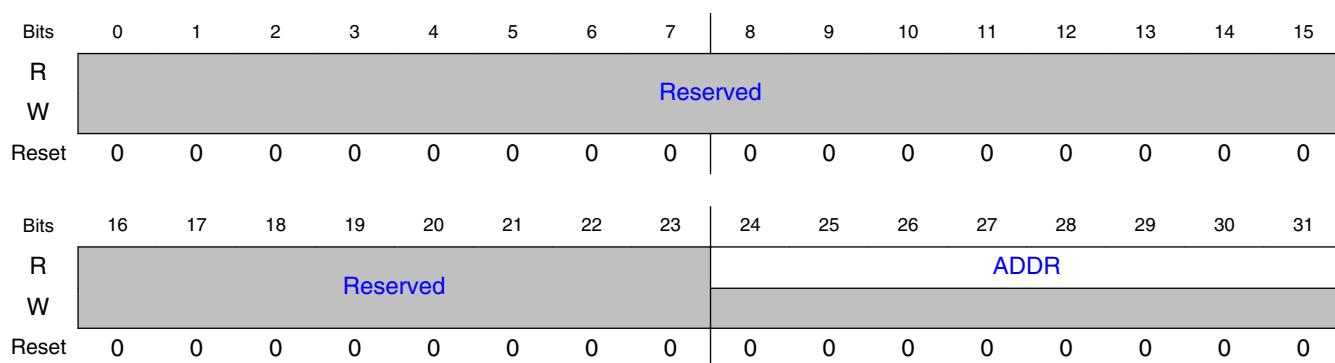
#### 31.2.1.37.2 Function

This is the command queue error capture address registers. When an enabled error condition occurs, additional information may be captured to indicate the source of the error. Software must clear the error condition to capture the next address.

The following error condition will capture the memory address:

Transaction error (CQEDR[TE]).

#### 31.2.1.37.3 Diagram



#### 31.2.1.37.4 Fields

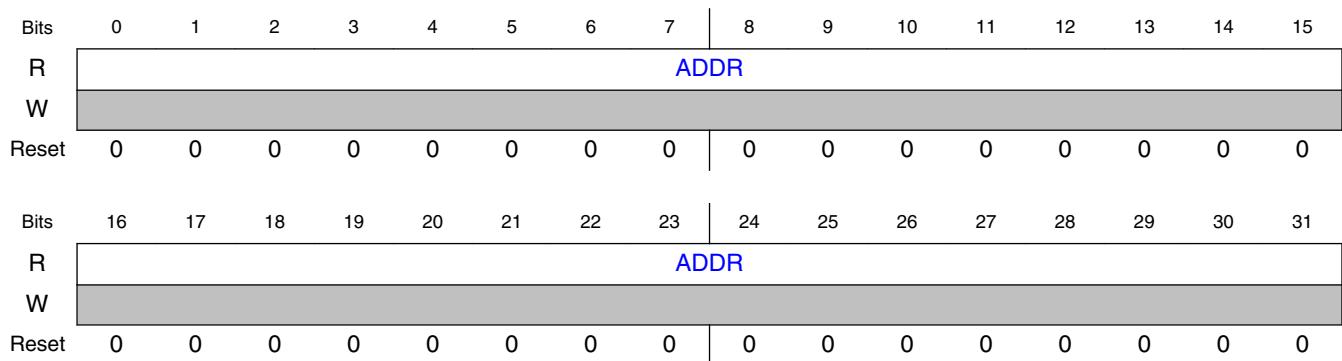
Field	Function
0-23	Reserved
—	
24-31	Address.
ADDR	

### 31.2.1.38 Command Queue Error Capture Address Register (CQECA R)

#### 31.2.1.38.1 Offset

Register	Offset
CQECAr	2_0A1Ch

#### 31.2.1.38.2 Diagram



#### 31.2.1.38.3 Fields

Field	Function
0-31 ADDR	Address.

### 31.2.1.39 Status queue critical congestion management register (SQCCMR)

#### 31.2.1.39.1 Offset

Register	Offset
SQCCMR	2_0A20h

### 31.2.1.39.2 Function

This register allows for setting of the critical watermark level for all status queues. The command queues will transition to XOFF if the critical level is reached.

The congestion watermark indicate how many available entries are left in the status queue. To avoid overflow of status queues, the watermark level should be set to accomodate for maximum number of in-flight jobs. The recommended value is 32. The critical watermark should always be at a higher level than the non-critical XOFF watermark.

### 31.2.1.39.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								ENTER_WM							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 31.2.1.39.4 Fields

Field	Function
0-9 —	Reserved
10-15 ENTER_WM	Critical congestion enter watermark level. This value indicates the number of status queue entries available. When the number of entries in the circular queue has reached the enter watermark, all command queues within the block transition to an XOFF state. For proper operation, this field should only be modified when the status queues are disabled.
16-31 —	Reserved

### 31.2.1.40 Command queue block a stream ID register (CQ0SIDR - CQ3SIDR)

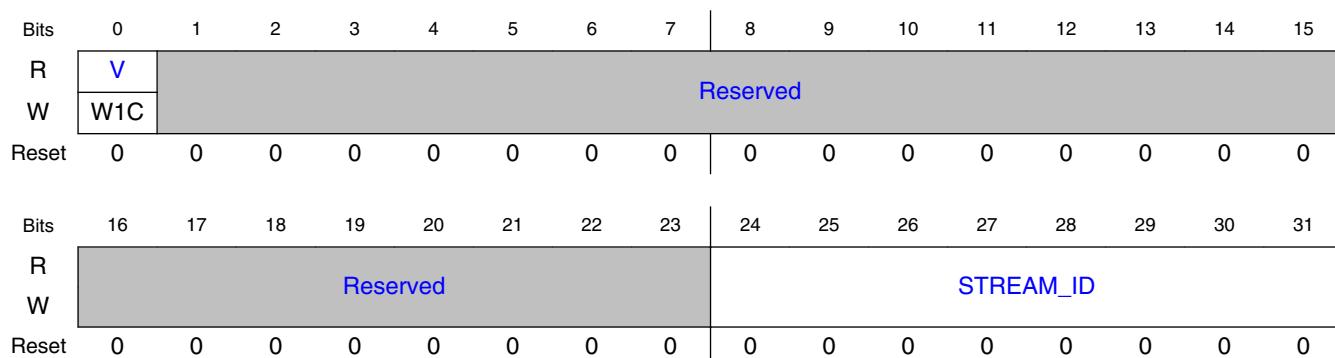
### 31.2.1.40.1 Offset

Register	Offset
CQ0SIDR	2_0A80h
CQ1SIDR	2_0A84h
CQ2SIDR	2_0A88h
CQ3SIDR	2_0A8Ch

### 31.2.1.40.2 Function

This is the command queue stream ID register for the block. It determines the access restrictions for all command descriptor and data transfers started using the command queues. Note that once the valid bit has been written, the stream ID can no longer be modified without hard reset.

### 31.2.1.40.3 Diagram



### 31.2.1.40.4 Fields

Field	Function
0 V	Valid. This is a sticky bit which can only be cleared through hard reset. When set, the stream ID can no longer be modified.
1-23 —	Reserved
24-31 STREAM_ID	Stream ID for data and descriptor accesses when using the command queues. Only writable when valid bit is 0.

## 31.3 Functional Description

The following sections describe the functionality of the qDMA controller in details.

### 31.3.1 qDMA Command Queue Mode Operation

Command queue mode is basically a support for circular command descriptor rings in memory. Software adds CDs at the head pointer and qDMA pulls CDs at the tail pointer.

#### 31.3.1.1 Dequeue Scheduling

Command queue scheduling refers to the selection of a command queue during a dequeue operation. Each block has 8 command queues, numbered 0-7. The scheduler is used to select the next command queue to be serviced as illustrated below.

**Table 31-3. Dequeue Scheduler**

Command Queue Number	Arbitration Group (Default)	First Level Scheduler	Second Level Scheduler
0	0	Weighted Interleaved Round Robin within an arbitration group.	Strict priority between arbitration groups. Lowest active arbitration group number wins.
1	1		
2	2		
3	3		
4	4		
5	5		
6	6		
7	7		

Arbitration group assignment rules:

1. Command queue 0 must be assigned AG0.
2. Arbitration group number assignment must be adjacent and increasing.
  - Example below shows two different arbitration groups assignments and scheduling applied. In the first example a 1-1-3-3 group structure is used,

utilizing all 8 queues. The second example uses a 2-2-1-1 arbitration group structure, where queue 6 and 7 are unused.

**Table 31-4. First Example**

Queue Number	Arbitration Group	First Level Scheduler	Second Level Scheduler
0	0	-	SP (Highest)
1	1	-	SP
2	2	WRR	SP
3	2		
4	2		
5	3	WRR	SP (lowest)
6	3		
7	3		

**Table 31-5. Second Example**

Queue Number	Arbitration Group	First Level Scheduler	Second Level Scheduler
0	0	WRR	SP (highest)
1	0		
2	1	WRR	SP
3	1		
4	2	-	SP
5	3	-	SP (lowest)
6	3	-	Unused
7	3	-	Unused

3. Arbitration group number assignment is uniform between blocks.
4. Unused queues in the system should be assigned the same AG number.

In the first level schedulers, Weighted Interleaved Round Robin (WIRR) is used to select one of  $n$  command queues (CQs) for service in each arbitration group. The relative weight (range 0 to 7) of each CQ is programmable, see Section 1.5.3.3, "Command

Queue Dequeue Scheduler Configuration Register 2 (CQDSCR2)". The number of selections made from each of the  $n$  queues in each group is tracked using a selection counter. The WIRR algorithm used within each arbitration group is summarized below:

- All selection counters for all arbitration groups start at 0.
- A queue is eligible for selection if it is non-empty and not flow controlled.
- An arbitration group is eligible for selection if there are eligible queues within that group.
- The lowest eligible arbitration group number is determined for each block of queues. A Round-Robin scheduler for the lowest arbitration group number selects the block.
- When a queue is selected for dequeue from the chosen block, the selection counter for the selected queue is incremented unless the counter has reached the programmed weight in which case the counter resets and the Round-Robin arbiter is updated.

Note that the scheduler's history registers (used for WIRR) are only updated when a scheduled dequeue from an arbitration group is performed and the programmed weight has been reached.

### 31.3.1.2 Status Queue

Status notification for DMA jobs are reported back to the status queue. Status information is carried within the command descriptor status/command field, bits 120-127. The command descriptor dequeue pointer advances only after the transaction has completed and the status information field has been updated.

The command descriptor address field will point to the command descriptor in its original format. It is the responsibility of the status queue consumer to deallocate buffers as needed when the command descriptor address pointer is non-zero.

### 31.3.1.3 qDMA Command Descriptor Formats

The qDMA accelerator supports compound command descriptor format.

#### 31.3.1.3.1 Compound Command Descriptor Format

**Table 31-6. qDMA Compound Command Descriptor Format**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
DD	—	QUEUE	—										ADDR										0x0C									
ADDR																													0x08			

*Table continues on the next page...*

**Table 31-6. qDMA Compound Command Descriptor Format (continued)**

FORMAT		OFFSET	—	0x04
—	S E R	—	STATUS	0x00

**Table 31-7. qDMA Compound Command Descriptor Field Descriptions**

Bits	Name	Description
127-126	DD	Dynamic debug marker. This field is intended for software use only and is returned as-is during enqueue of the descriptor. The purpose being debug or marking of specific commands.
125-123	-	Reserved
122-120	QUEUE	Command queue origin produced by qDMA, intended for consumer of command descriptor.
119-104	-	Reserved
103-64	ADDR[0:39]	40-bit address. The memory address indicating the start of the buffer holding the compound frame table describing the frame.
63-61	FORMAT	Frame format. A coded value which defines the format of the frame referenced by this FD. 001 Compound S/G format All other values reserved.
60-52	OFFSET	Frame offset. Frame offset is the number of bytes from the frame address (ADDR) at which actual data begins. If a frame carries a non-zero offset, additional information may be stored in the buffer carrying the frame, from ADDR to ADDR + OFFSET - 1. For compound frame format this field has to be zero.
51-32	-	Reserved
31	-	Reserved
30	SER	Status notification enqueue required. 0 No status notification is enqueued. 1 Status notification is enqueued to status queue.
29-8	-	Reserved
7-0	STATUS	Command status. Information to the command descriptor consumer indicating the outcome of the DMA operation. Please see the global error registers for additional information. For bit settings please refer <a href="#">Table 31-8</a>

**Table 31-8. Bit setting for STATUS**

Bits	Name	Description
7	-	Reserved
6	-	Reserved
5	RTE	Read transaction error. Transaction error during data read.
4	WTE	Write transaction error. Transaction error during data write.
3	-	Reserved
2	CDE	Command descriptor error. A format error was detected when parsing the command descriptor. This includes: <ul style="list-style-type: none"><li>• Invalid CD format option</li></ul>

*Table continues on the next page...*

**Table 31-8. Bit setting for STATUS (continued)**

1	SDE	Source descriptor error. This includes: <ul style="list-style-type: none"><li>• Zero length,</li><li>• Zero stride size when SD[SSEN]=1</li><li>• Illegal SD[SRTYPE].</li></ul>
0	DDE	Destination descriptor error. This includes: <ul style="list-style-type: none"><li>• Illegal length (destination length exceeds source length)</li><li>• Illegal DD[DWTYPE].</li><li>• S/G table early termination</li></ul>

### 31.3.1.4 The qDMA Compound S/G Format

The figure below shows the compound S/G format used by the qDMA accelerator.

**Table 31-9. qDMA Compound S/G Table Entry Format**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
																													ADDR	0x0C		
																													ADDR	0x08		
E	F	LENGTH																											0x04			
																													OFFSET	0x00		

**Table 31-10. qDMA Compound S/G Field Descriptions**

Bits	Name	Description
127-104	-	Reserved
103-64	ADDRESS	Compound source/destination entry address (40-bits). It is advantageous for performance reason but not required that address be 64B aligned.  The compound entry address points to either scatter/gather table or data buffer depending on E bit. <ul style="list-style-type: none"><li>• E=0, this address points to data buffer.</li><li>• E=1, this address points to scatter/gather table.</li></ul>
63	E	The extension bit.  0 The entry references to a single buffer.  1 The entry references to a scatter/gather table. Not allowed for the first entry which holds the source and destination descriptors.
62	F	The final bit  0 There are one or more entries after this entry in the compound S/G.  1 This entry is the last entry in the compound S/G.

*Table continues on the next page...*

**Table 31-10. qDMA Compound S/G Field Descriptions (continued)**

Bits	Name	Description
61-32	LENGTH	This is the total number of data bytes in source or each destination. Note that destination data length can be 0, in which case, qDMA will skip this destination. When destination data length > source data length, destination descriptor error is reported. When source data length equal 0, source descriptor error is reported.  The first compound S/G entry must point to the source descriptor, followed by destination descriptor. Only single data buffer is supported for descriptors, i.e. CF[E]=0. The length must be 32B. A descriptor error is flagged if any of the rules above are not followed.  If striding is enabled, the destination may write less than LENGTH data bytes for source and/or destination.
31-13	-	Reserved
12-0	OFFSET	The number of bytes offset from the ADDRESS at which the data starts.

### 31.3.1.5 The qDMA Source Descriptor Format

The figure below shows the source descriptor format used by the qDMA accelerator when using compound S/G.

**Table 31-11. qDMA Source Descriptor Format**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CMD																													0x0C			
—																													0x08			
—																														0x04		
—																															0x00	

**Table 31-12. qDMA Source Descriptor Field Descriptions**

Bits	Name	Description
127-96	CMD	Source descriptor command. Additional information from the producer of the command descriptor. For field definitions please, <a href="#">Table 31-13</a>
95-56	-	Reserved
55-44	SSS	Source stride size. This is the number of bytes to transfer before jumping to the next start byte as specified in the SSD field. If the stride size is larger than the data buffer transfer size, a source descriptor error will occur.
43-32	SSD	Source stride distance. The stride distance in bytes from the start byte to the next start byte. When the stride distance is 0, the qDMA performs source address hold function. To guarantee a single transaction when using hold function, the stride size must be limited to 1, 2, 4 or 8 bytes aligned to the size.
31-0	-	Reserved

**Table 31-13. Field definitions for CMD**

Bits	Name	Description
31-28	SRTTYPE	DMA source read transaction type, see <a href="#">Table 31-14</a>
27	NS	Data no-snoop. 0 Snoop 1 No-snoop
26-24	SQOS	Source read transaction QoS. qDMA does not use this for internal arbitration, it is only passed along with the transaction for possible quality of service at other arbitration/switch points in the path to the source. 000 Lowest quality of service 001 Next lowest quality of service ... 110 Next highest quality of service 111 Highest quality of service
23-20	RTHROTL	Read throttle control. Minimum number of platform cycles to accumulatively issue 256 bytes of read data. The qDMA engine resumes automatically when RTHROTL number of platform cycles has elapsed since issuing the first read in this 256-byte batch.
		0000 Disabled. 0001 256 cycles 0010 512 cycles 0011 1024 cycles 0100 2048 cycles
		0101 4096 platform cycles 0110 8192 platform cycles 0111 16,384 platform cycles 1000-1111 Reserved
19	SSEN	Source stride enable.
18	-	Reserved
17	PF	Prefetchable. 0 Non-prefetchable. 1 Prefetchable.
16-0	-	Reserved

The table below describes the SRTTYPE encodings.

**Table 31-14. SRTTYPE Encodings**

Encodings	Local Access
0000	Reserved
0001	Reserved
0010	Reserved
0011	Reserved
0100	Read
0101	Reserved
0110	Reserved
0111	Reserved

*Table continues on the next page...*

**Table 31-14. SRTTYPE Encodings  
(continued)**

1000	Reserved
1001	Reserved
1010	Reserved
1011	Reserved
1100	Reserved
1101	Reserved
1110	Reserved
1111	Reserved

### 31.3.1.6 The qDMA Destination Descriptor Format

The figure below shows the destination descriptor format used by the qDMA accelerator when using compound S/G.

**Table 31-15. qDMA Destination Descriptor Format**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
CMD																														0x0C		
—																													0x08			
—																														0x04		
—																															0x00	

**Table 31-16. qDMA Destination Descriptor Field Descriptions**

Bits	Name	Description
127-96	CMD	Destination descriptor command. Additional information from the producer of the command descriptor. For field definitions please refer to <a href="#">Table 31-17</a>
95-64	-	Reserved
63-56	-	Reserved
55-44	DSS	Destination stride size. This is the number of bytes to transfer before jumping to the next start byte as specified in the DSD field. If the stride size is larger than the data buffer transfer size, a destination descriptor error will occur.
43-32	DSD	Destination stride distance. The stride distance in bytes from the start byte to the next start byte. When the stride distance is 0, the qDMA performs destination address hold function. To guarantee a single transaction when using hold function, the stride size must be limited to 1, 2, 4 or 8 bytes aligned to the size.
31-0	-	Reserved

**Table 31-17. Field Definitions for CMD[31:0]**

Bits	Name	Description
31-28	DWTTYPE	DMA destination write transaction type, see <a href="#">Table 31-18</a>
27	NS	Data no-snoop. 0 Snoop 1 No-snoop
26-24	DQOS	Destination write transaction QoS. qDMA does not use this for internal arbitration, it is only passed along with the transaction for possible quality of service at other arbitration/switch points in the path to the destination.  000 Lowest quality of service 001 Next lowest quality of service ... 110 Next highest quality of service 111 Highest quality of service
23-20	WTHRCTL	Write throttle control. Minimum number of platform cycles to accumulatively issue 256 bytes of write data. The qDMA engine resumes automatically when WTHRCTL number of platform cycles has elapsed since issuing the first write in this 256-byte batch.  0000 Disabled. 0001 256 cycles 0010 512 cycles 0011 1024 cycles 0100 2048 cycles 0101 4096 platform cycles 0110 8192 platform cycles 0111 16,384 platform cycles 1000-1111 Reserved
19	DSEN	Destination stride enable.
18	-	Reserved
17-16	LWC	Last write control.  00 The last write is a non-posted write.  01 The last write is a posted write chased with a prefetchable read. The read size is the same as last write when write size < 4B or 4B otherwise.  10 The last write is a posted write chased with a non-prefetchable read. The read size is the same as last write when write size < 4B or 4B otherwise.  11 The last write is a posted write.  Notes: Reads to volatile memory should be avoided since there may be side effects.
15	-	Reserved
13	-	Reserved
12-0	-	Reserved

The table below describes the DWTYPE encodings.

**Table 31-18. DWTYPE Encodings**

Encodings	Local Access
0000	Reserved
0001	Reserved
0010	Reserved
0011	Reserved
0100	Write
0101	Reserved
0110	Reserved
0111	Reserved
1000	Reserved
1001	Reserved
1010	Reserved
1011	Reserved
1100	Reserved
1101	Reserved
1110	Reserved
1111	Reserved

### 31.3.1.7 The qDMA Scatter Gather Table Format

Each compound command entry, except the first descriptor entry, may use the 16-byte scatter/gather (S/G) table format for data management. The extension bit indicates if the address/offset fields are pointing to a direct data memory location or another S/G table. The definition of a S/G table entry is described here. Each S/G table contains one or more entries, and may be extended to include additional tables. Each S/G entry points to a data buffer or chains to another S/G table as required.

Each S/G entry has a 40-bit address field and a 30-bit buffer length field. The E (extension) bit determines if the address is pointing to a data buffer (E=0) or to a new S/G table (E=1). The F (final) bit determines when the last entry of the table has been reached.

**Table 31-19. Scatter/Gather Table Entry Format**

31	30	29	28	27	26	25	24	23	22	21	20	1 9	18	17	1 6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset
																										ADDR		0x0C				
																													0x08			
E	F	LENGTH																											0x04			

*Table continues on the next page...*

**Table 31-19. Scatter/Gather Table Entry Format (continued)**

—	OFFSET	0x0C
---	--------	------

**Table 31-20. Scatter/Gather Table Entry Field Descriptions**

Bits	Name	Description
127-104	-	Reserved
103-64	ADDRESS	40 -bit byte address to data buffer or discontinued S/G table entry, depending on E bit setting.
63	E	<p>Extension.</p> <p>0 Address references a data buffer.</p> <p>1 Address references a S/G table.</p> <p>The extension bit has precedence over the final bit. If both are set the final bit is ignored.</p>
62	F	<p>Final entry.</p> <p>0 Not last table entry.</p> <p>1 Indicates final table entry.</p> <p>This bit is ignored if the extension bit is set.</p>
61-32	LENGTH	Effective length of the data buffer when E=0, ignored when E=1.
31-13	-	Reserved
12-0	OFFSET	Byte offset from ADDRESS where data payload or S/G table reside.

A buffer length of zero signifies that no data is transferred for that S/G segment and the address should not be accessed. It does not signify end of list nor has any other special meaning. The length is ignored if the extension bit is set. Only the last S/G segment may chain by setting the extension bit, it does not have to chain. The offset value may be non-zero to indicate significant data is stored there.

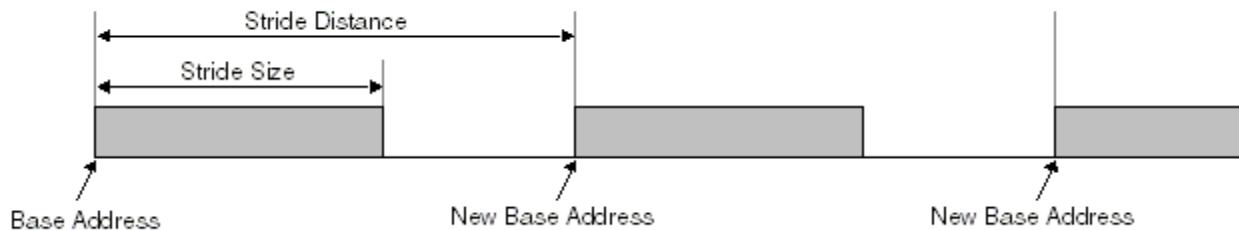
The overall data payload length as defined by the compound command entry LENGTH field is also used to determine the end of a S/G table. If the length has been reached, regardless of the final bit setting, the last segment has been reached.

Note that while scatter/gather tables may offer better memory utilization, they can potentially decrease usable data bandwidth and increase startup latency due to an increase in prefetches by qDMA to handle the added indirection.

### 31.3.1.8 Striding Operation

If operating in striding mode, SD[SSE]=1 and/or DD[DSE]=1, the stride size defines the amount of data to transfer before jumping to the next quantity of data as specified by the stride distance. The stride distance is added to the current base address to point to the next quantity of data to be transferred. [Figure 31-2](#) illustrates the stride size and distance parameters. As shown, each time the stride distance is added to the base address, the

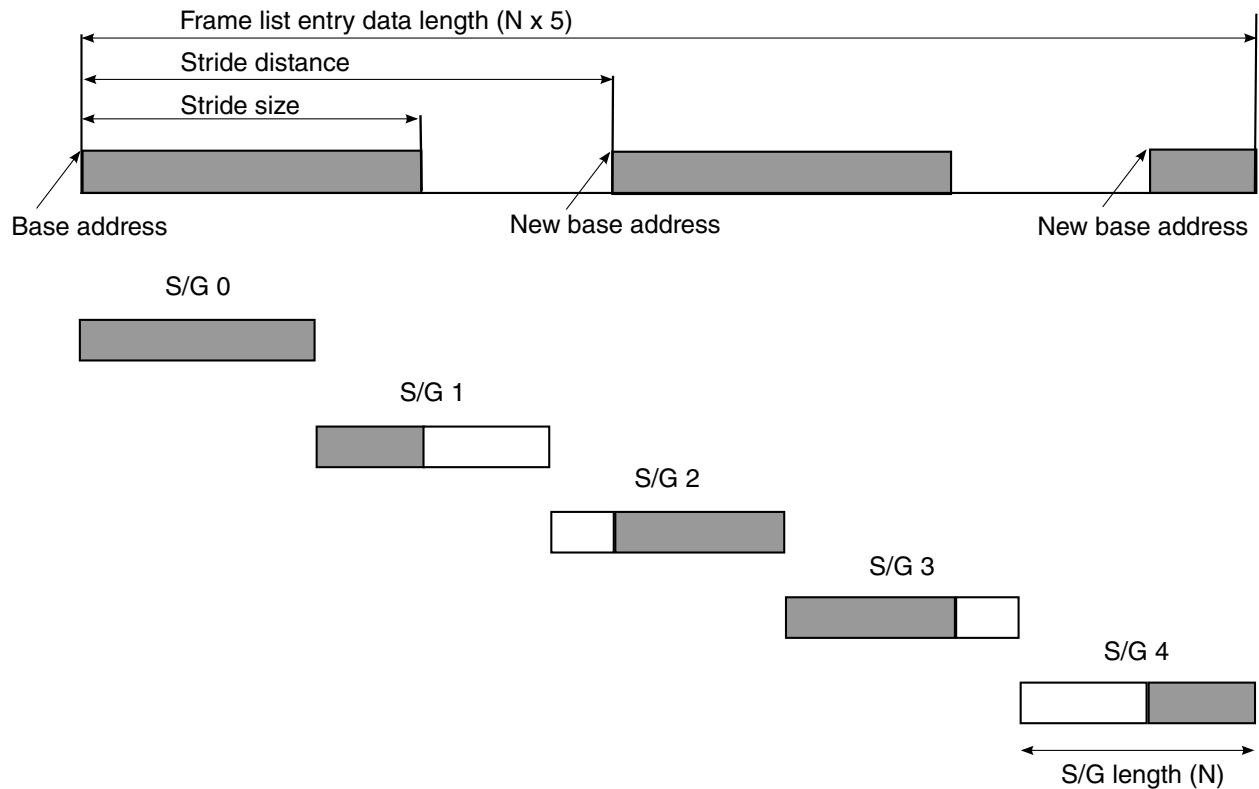
resulting address becomes a new base address. This sequence repeats until the end of the data buffer has been reached. The stride distance has to be zero (hold) or equal/greater than the stride size, or a descriptor error is set.



**Figure 31-2. Stride Size and Stride Distance**

When the stride distance is 0, the qDMA performs an address hold operation. This operation is only allowed in non-S/G mode. qDMA will read or write as many bytes as defined by the compound frame length.

When striding across a single data buffer, the buffer size is indicated by the compound command length. Striding continues until the end of the buffer is reached. Less bytes may actually be read/written if the stride distance is greater than the stride size. Same holds true for S/G format, where the buffer size is determined by the S/G length. The result of striding is the same if striding across a single large buffer or a S/G representation of the same. The figure below shows an example of striding across a large single data buffer, and the same result if the data buffer was represented as a S/G list. Note that if source striding is enabled, the destination can only write as many bytes as being read, thus the destination will only write up to the lesser of compound command length or actual data read.



**Figure 31-3. Striding Across Single Data Buffer or S/G Data Buffers**

### 31.3.1.8.1 Summary of Striding Rules

The following striding rules apply.

1. Stride distance has to be zero (hold function) or greater/equal to stride size.
2. When striding distance is zero (hold function), S/G format is not supported.
3. When source striding is enabled (SD[SSE]=1), the destination can only write as many bytes as are read from the source.

### 31.3.1.9 Partition Level Reset Assistance

The qDMA allows privileged software to reset or flush pending and new DMA jobs matching a stream ID. The following sequence is recommended for this operation:

1. Logical partition N is disabled from enqueueing further jobs to qDMA. How this is accomplished is beyond this specification.

2. Privileged software initializes DSIDFR[STREAM\_ID] with the logical partition requiring reset assistance and enables the DMA stream ID flush register, DSIDFR[FM]=b01.
3. qDMA will terminate any DMA job using the same stream ID specified in DSIDFR[STREAM\_ID].
  - Pending DMA jobs will terminate at earliest convenient time with no status notification.
  - New DMA jobs dequeued from the direct portal will terminate immediately with no status notification.
4. When privileged software has determined no more jobs are available for the partition, the flush command is disabled by setting DSIDFR[FM]=0b00.

### 31.3.1.10 Different Use Cases

#### 31.3.1.10.1 Single Destination with Single Source/Destination Data Buffer

In case of single destination, there are three entries in the compound command, one descriptor, one source data entry and one destination data entry with final bit set to 1. When single data buffer is used, the extension bit in the compound command needs to be 0.

The figure here shows an example format for this use case.

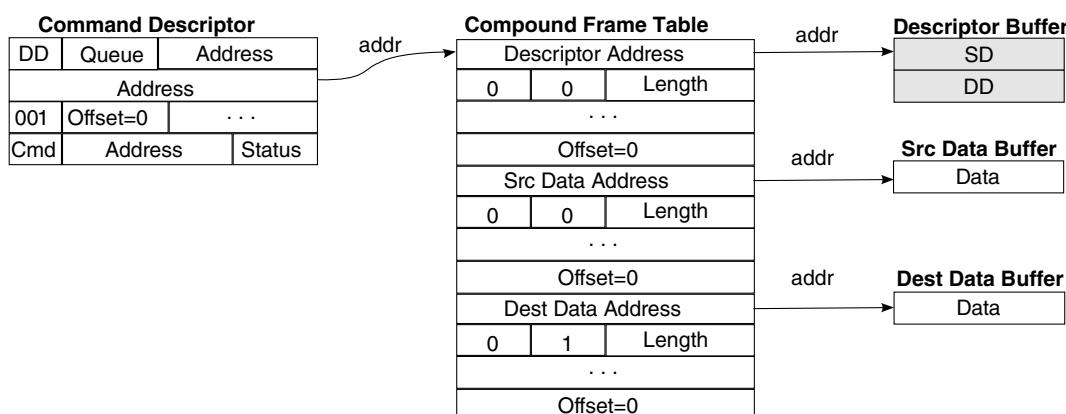
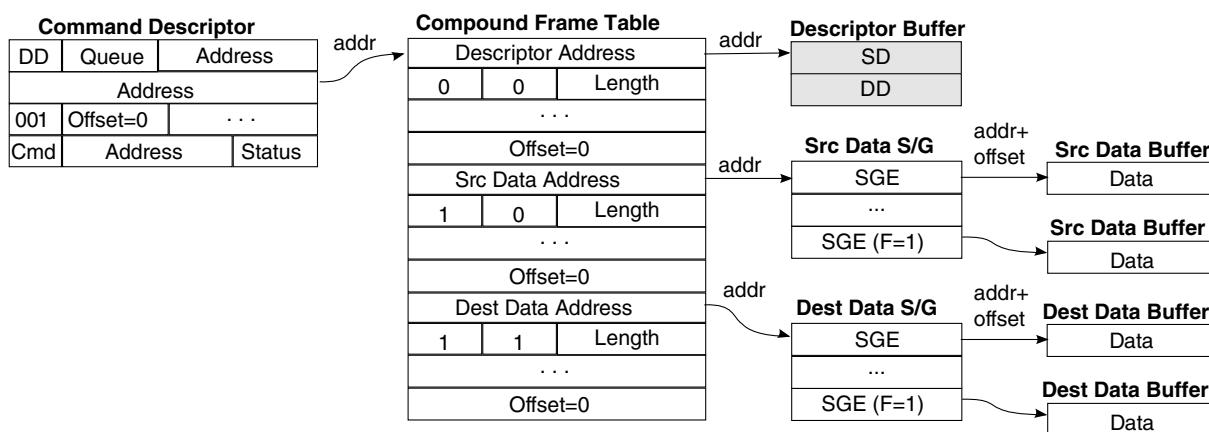


Figure 31-4. Single destination with single data buffers format

### 31.3.1.10.2 Single Destination with Multiple Data Buffers

In case of single destination, there are only three entries in the compound command, one descriptor entry, one source data entry and one destination entry, with final bit set to 1. When multiple data buffer are used, extension bit in the compound command entry is set to 1. Each scatter/gather table entry points to a data buffer (E=0) and the last entry may be further extended as needed (E=1).

The figure here shows an example format for this use case.



**Figure 31-5. Single destination with multiple data buffers format**

### 31.3.2 qDMA Controller Interrupts

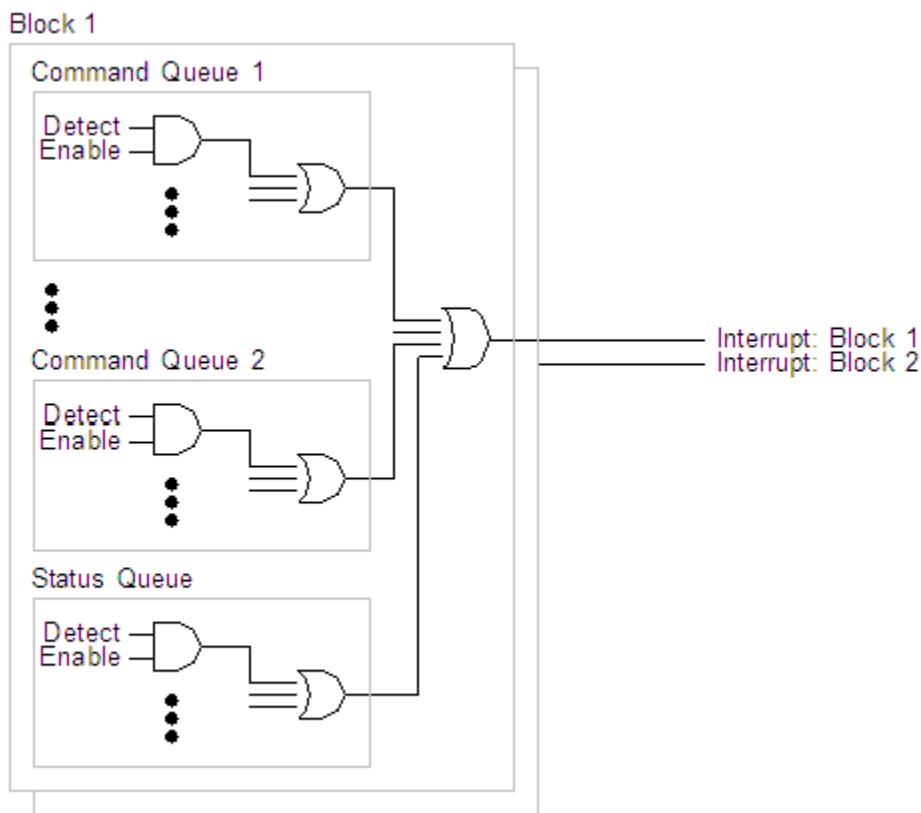
The qDMA controller can only generate error interrupts. Interrupt detect from DEDR is qualified with its corresponding interrupt enable from DEIER. The result is combined to form one error interrupt.

### 31.3.3 Command Queue Interrupts

There are two types of interrupts: error and functional status. Only one error interrupt is generated, but there is one functional interrupt per block.

Each interrupt detect from CQEDR is qualified with its corresponding interrupt enable from CQIER. The result of each is combined to form one error interrupt.

The figure below shows the functional interrupt structure. These interrupts correspond to the command and status queues. Each interrupt detect from BaCQbIDR is qualified with its corresponding interrupt enable from BaCQbIER. The result of each is combined to form one functional interrupt per block.



**Figure 31-6. Functional Interrupt Structure**

### 31.3.4 Global Bandwidth Throttling

Global bandwidth throttling allows software to control the maximum bandwidth amount given to qDMA in the system. DMA transactions are often bursty in nature and may impact performance of other devices also competing for bandwidth over a shared fabric. By default this feature is disabled, allowing qDMA to consume as much bandwidth as needed.

The qDMA controller supports global bandwidth throttling through a token bucket bandwidth allocation scheme. This allows the qDMA controller to obtain tokens which are traded for bandwidth usage (bytes) on the system to smooth out bursty transactions which can affect other devices in the system negatively by resource starvation. Global bandwidth throttling is enabled by setting DMR[GBT]=1. When disabled, the qDMA controller has unlimited bandwidth quota.

The registers DGBTR controls the bandwidth sharing scheme to set the token bucket addition rate and capacity. The addition rate is the time it takes to add one token (or byte) to the token bucket counter. The maximum number of tokens that can be accumulated is set in the token capacity register. Any tokens accumulated above the token bucket capacity are lost. The following conditions will stop the qDMA controller from issuing any read or write transactions:

- The capacity is less than 256 bytes.
- The number of available tokens are less than the transaction size

The largest read or write transaction issued by the qDMA controller is 256 bytes, hence the capacity must be set to a value equal to or greater to this value. Here are some guidelines of controlling the bandwidth throttling:

- A large token capacity can still result in bursty transactions. It is better to have a medium sized token capacity and adjust the addition rate to smooth out bandwidth usage.

### **31.3.5 Address Alignment Requirements**

qDMA supports byte alignment for all descriptor and data read/writes.

### **31.3.6 Transaction Sizes**

qDMA generates transactions with a maximum transaction size of 256 bytes with any alignment. Transactions that overlap a 4K byte address boundary are split into two transactions.

### **31.3.7 Performance Improvements**

#### **31.3.7.1 Large Transfers**

In case of large or long latency transfers, overall DMA performance may be improved if the job is broken up in to smaller jobs. DMA processing engines are virtualized to handle a large number of job producers. A long latency job in its entirety has the potential to occupy hardware resources for a long time, so breaking such jobs into smaller jobs allows for improved scheduling by the DMA controller.

### 31.3.7.2 Alignment

The following software actions may improve performance:

- Align transactions to the memory sub-system if less than 256 bytes. This will avoid transactions splitting unnecessarily.
- Non-prefetchable read setting in the source descriptor may be required for targets other than local memory. Prefetchable read setting will offer better performance for misaligned transfers in the form of fewer transactions and should be set if possible.

### 31.3.8 Low-Power Mode

In order to enter low-power mode, qDMA needs to be in an idle state. qDMA will only indicate idle state if no jobs are currently being processed. Software must set DMR[DQD]=1 to stop further dequeue of command descriptors, before requesting low-power mode.

### 31.3.9 DMA Scenarios

The following is a description of unusual DMA paths including explanations of why some functional blocks cannot serve as qDMA targets. The following topics are addressed:

- DMA transaction initiators (masters)
- DMA targets, that is, data sources or destinations
- Transparency of the bus controllers to DMA transactions

What is useful as opposed to what is possible. For example, any register can be addressed through an internal control bus, which means configuration and control registers can be DMA targets.

#### 31.3.9.1 qDMA to Core

The L1 cache cannot be a direct DMA target because it cannot be directly addressed by software. However, DMA access into the L1 cache occurs indirectly if a block of memory that is cached in the L1 is specified as the DMA target.

### 31.3.9.2 qDMA to Configuration, Control and Status Registers

Because any internal register can be addressed with the qDMA controller, configuration, control, and status registers throughout the device are valid DMA targets. However, the primary purpose of DMA-to reduce processor load by moving large blocks of data-is not served by DMA transfers of configuration data. For example, while it is possible to DMA into the I2C controller or programmable interrupt controller (PIC), doing so is extremely inefficient and is seldom beneficial in normal operation. The overhead of creating DMA descriptors far exceeds any savings in CPU cycles.

### 31.3.9.3 qDMA to PCI-Express

As a DMA target, PCI-Express does not support non-posted writes. Thus it is not possible based on the last write transaction to determine if/when the actual write has occurred on PCI-Express target. The qDMA controller therefore supports a mode in the destination descriptor, DD[LWC], to allow the last write to be channed with a read operation.

## 31.3.10 Error Handling

There are two types of errors, hardware and programming errors. The error checking level indicates the order in which errors are checked. Multiple errors can be detected at an error checking level. Once an error is detected, no additional error checking beyond the current level is performed. The first error detected in the processing pipeline updates the error capture registers.

### 31.3.10.1 Hardware Errors

The following tables list hardware error detected by the qDMA controller.

**Table 31-21. qDMA Command Queue Mode Hardware Errors**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
Read transaction error.	1,3	qDMA error interrupt if DEIER[RTEIE] is set.	Read transaction error, DEDR[RTE] and CD[RTE]	Command descriptor, command queue ID, and byte count in global registers.  Byte count transferred and target number in command descriptor.
Comments				

*Table continues on the next page...*

**Table 31-21. qDMA Command Queue Mode Hardware Errors (continued)**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
	Uncorrectable ECC errors. Software should query the source and target for error status information and take appropriate action. qDMA reports internal ECC errors to the system.			
Write transaction error.	3	qDMA error interrupt if DEIER[WTEIE] is set.	Write transaction error, DEDR[WTE] and CD[WTE]	Command descriptor, command queue ID, and byte count in global registers.  Byte count transferred and target number in command descriptor.
	Comments			
	Uncorrectable ECC errors. Software should query the source and target(s) for error status information and take appropriate action. qDMA reports internal ECC errors to the system.			

### 31.3.10.2 Programming Errors

The following tables list programming errors detected by the qDMA controller and command queues. In summary, programming errors are reported in the following ways:

1. Enqueue of status notification command descriptor for transactions operating in command queue mode
 

A programming error is reported in the command descriptor status field.
2. Global error reporting for transactions operating in command queue mode

Programming errors are reported globally in the error detect register, DEDR, for all DMA engines when encountered. Error capture registers may hold additional information about the error condition. If status notification is required, it will always be reported to the source regardless of error type.

**Table 31-22. qDMA Command Queue Mode Programming Errors**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
Read transaction error.	1,3	qDMA error interrupt if DEIER[RTEIE] is set.	Read transaction error, DEDR[RTE] and CD[RTE]	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	Address mapping errors.			
Data write transaction error.	3	qDMA error interrupt if DEIER[WTEIE] is set.	Write transaction error, DEDR[WTE] and CD[WTE]	Command descriptor, command queue ID, and byte count in global registers.

*Table continues on the next page...*

**Table 31-22. qDMA Command Queue Mode Programming Errors (continued)**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture	
		Comments			
		Address mapping errors.			
Command descriptor FORMAT is not 0b001.	2	qDMA error interrupt if DEIER[CDEIE] is set	Command descriptor error, DEDR[CDE] and CD[FDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Unsupported frame format. Only the compound frame format (0b001) is supported.			
Command descriptor OFFSET is not zero when FORMAT is 0b001.	2	qDMA error interrupt if DEIER[CDEIE] is set	Command descriptor error, DEDR[CDE] and CD[FDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Unsupported frame offset. The frame descriptor offset must be zero for compound frame format.			
Status notification enqueue rejection.	4	qDMA error interrupt if DEIER[EREIE] is set	Enqueue Rejection Error, DEDR[ERE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Status notification enqueue is rejected by QMAN. Further enqueue may be halted until software clears DEDR[ERE].			
Unknown source descriptor SRTTYPE.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Unknown source read transaction type, when ATMU bypass specified.			
Unknown source descriptor target interface STINT.	3	qDMA error interrupt if DEIER[SDEIE] is set	Destination Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Unknown source read target interface when ATMU bypass specified.			
Striding enabled and source stride size is zero.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			
		Nothing to transfer.			
Striding enabled and source stride size is greater than source stride distance.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.	
		Comments			

*Table continues on the next page...*

**Table 31-22. qDMA Command Queue Mode Programming Errors (continued)**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
	The actual number of bytes to read from a data buffer is ambiguous.			
Striding enabled and source stride distance is zero when compound command indicates S/G table.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and DD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	When stride hold function is used, striding across multiple buffers is not allowed.			
Compound command descriptor entry final (F) bit or extend (E) bit are 1.	2	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	S/G format is not supported for source and destination descriptor buffer.			
Compound command descriptor length not 32 bytes.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	The compound command descriptor entry length field must be 32 bytes in size to reflect one source descriptor and one destination descriptor.			
The compound command source data entry LENGTH is 0.	2	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	Nothing to transfer. Software should not rely on a zero length source as a no-operation since there may be unforeseen side-effects to this error. No transactions are issued by qDMA.			
Compound command source entry final (F) bit is 1	2	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	Nothing to transfer. The compound frame must contain at least 3 entries (one destination), see <a href="#">Different Use Cases</a> .			
Compound command source entry offset non-zero when extend (E) bit is 1.	2	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			
	S/G table cannot start at a non-zero offset.			
Source scatter/gather total length smaller than compound command source entry LENGTH.	3	qDMA error interrupt if DEIER[SDEIE] is set	Source Descriptor Error, DEDR[SDE] and CD[SDE].	Command descriptor, command queue ID, and byte count in global registers.
	Comments			

Table continues on the next page...

## Functional Description

**Table 31-22. qDMA Command Queue Mode Programming Errors (continued)**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
	Source data buffer is not large enough to contain source data. The compound frame LENGTH field should be equal or less than the source S/G data length.			
Striding enabled and destination stride size is zero.	3	qDMA error interrupt if DEIER[DDEIE] is set	Destination Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments Nothing to transfer.				
Striding enabled and destination stride size is greater than destination stride distance.	3	qDMA error interrupt if DEIER[DDEIE] is set	Source Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments The actual number of bytes to write to a data buffer is ambiguous.				
Striding enabled and destination stride distance is zero when compound command indicates S/G table.	3	qDMA error interrupt if DEIER[DDEIE] is set	Source Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments When stride hold function is used, striding across multiple buffers is not allowed.				
Compound command destination entry LENGTH is greater than compound command source entry LENGTH.	3	qDMA error interrupt if DEIER[DDEIE] is set	Destination Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments Destination data would be unknown. If source striding is disabled, the error occurs before any data is written. If source striding is enabled, a runtime error will occur after all data read has been written.				
Compound command destination entry offset non-zero when extend (E) bit is 1	3	qDMA error interrupt if DEIER[DDEIE] is set	Source Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments S/G table cannot start at a non-zero offset.				
Destination scatter/gather total length smaller than compound command destination entry LENGTH	3	qDMA error interrupt if DEIER[DDEIE] is set	Destination Descriptor Error, DEDR[DDE] and CD[DDE].	Command descriptor, command queue ID, and byte count in global registers.
Comments Destination data buffer is not big enough to contain destination data. This may be due to early termination of S/G list.				
Incorrect mode combination when FD[MM]=1	2	qDMA error interrupt if DEIER[DDEIE] is set	Destination Descriptor Error, DEDR[DDE] and FD[DDE].	Frame descriptor, frame queue ID, and byte count in global registers.
Comments				

*Table continues on the next page...*

**Table 31-22. qDMA Command Queue Mode Programming Errors (continued)**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
	Inter-partition message mode requires DD[BAR]=1, FD[SER]=0b1x, and compound frame destination entry E bit is 0.			

The table here shows programming errors related to the command queues, which is separate from the qDMA controller.

**Table 31-23. qDMA Command Queue Programming Errors**

Error	Error Checking Level	Interrupt Generated	Status Bit Set	Error Capture
Transaction error.	1,4	Command queue error interrupt if CQIER[TEIE] is set.	Transaction error, CQEDR[TE].	Address error capture registers, CQEC(E)AR.
Comments Address mapping errors. Command queue responds with an empty command descriptor for dequeue. The enqueue pointer for writes or dequeue pointer for reads is not updated				
Programming error.	4	Command queue interrupt if BaCQIER[CQPEIE] is set.	Programming error, BaCQIDR[SQPE]	None.
Comments A programming error indicates that a command descriptor push was attempted on a disabled status queue.				

## 31.4 Initialization Information

qDMA only supports command queue mode, which indicates the source of the transactions. Privileged space registers control access restrictions and operational modes, while manager space registers involve resource management.

**Table 31-24. Initialization Information**

Function	Register(s)	Mode(s)	Control
Recommended Configuration			
Disable the command queue, BaCQbMR[EN]=0	DMR[DQD]=1 DSR[DB]=0	Command Queue	Block
Disable dequeuing of command descriptor from command queues and wait for qDMA idle state.	DMR[DQD]=1 DSR[DB]=0	Command Queue	Privileged
Disable the status queues.	BaSQMR[EN]=0	Command Queue	Block

*Table continues on the next page...*

**Table 31-24. Initialization Information (continued)**

Function	Register(s)	Mode(s)	Control
Set command queue block stream ID for DMA transfer access control.	CQBnSIDR	Command Queue	Privileged
Set the command queue size field.	BaCQbMR[CQ_SIZE]	Command Queue	Block
Set the status queue size field.	BaSQMR[CQ_SIZE],	Command Queue	Block
Initialize the command queue pointers to match. Must be aligned to queue size. The command queue detects address pointer wrap conditions only when enabled. If the pointers are unequal when the queue is enabled, no wrap condition must exist.	BaCQn(E)DPAR BaCQn(E)EPAR	Command Queue	Block
Initialize the status queue pointers to match. Must be aligned to queue size.	BaSQ(E)DPAR BaSQ(E)EPAR	Command Queue	Block
Clear the command queue interrupt detect register bits for all queues.	BaCQIDR	Command Queue	Block
Clear the status queue interrupt detect register bits for all queues.	BaSQIDR	Command Queue	Block
Clear qDMA controller error detect register bits.	DEDR	Command Queue	Manager
Set error interrupt enable register, if interrupt is required as a result of an error condition.	DEIER	Command Queue	Manager
Enable the command queues.	BaCQbMR[EN]=1	Command Queue	Block
Enable the status queues.	BaSQMR[EN]=1	Command Queue	Block
Enable qDMA controller dequeue of command descriptor from command queues.	DMR[DQD]=0	Command Queue	Privileged
Add a single command descriptor to the command queue and increment the enqueue pointer by setting the enqueue increment bit or for multiple descriptors write the enqueue address register directly accounting for wrap condition.	BaCQbMR[EI] or BaCQb(E)EPAR	Command Queue	Block
Optional Settings			
DMA global bandwidth throttling if system bandwidth sharing is required.	DGBTR, DMR[GBT]	Both	Privileged
Global overrides of transaction attributes, i.e. snoop.	DMR	Both	Privileged
Prefetch quality of service.	DMR	Command Queue	Privileged

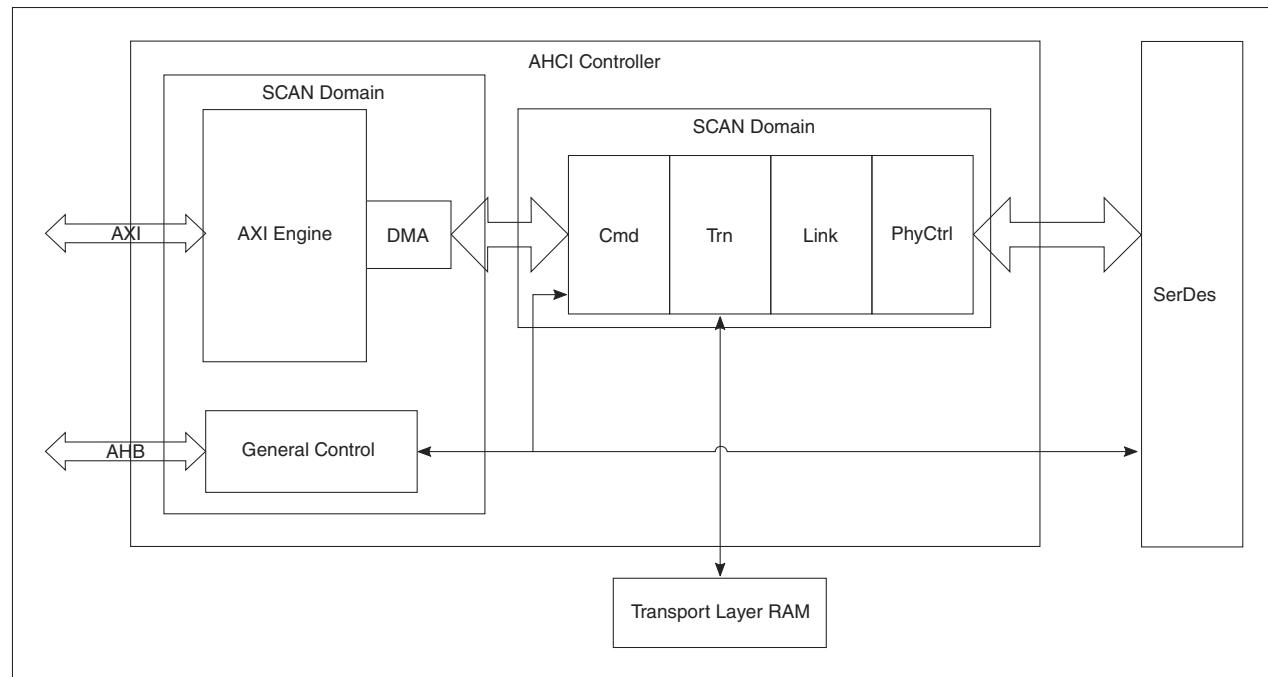
# **Chapter 32**

## **SATA 3.0**

### **32.1 Advanced host controller interface overview**

The SATA 3.0 advanced host controller interface (AHCI) is a high-performance SATA solution that delivers comprehensive and fully-compliant generation 3 (1.5 Gb/s - 6.0 Gb/s) serial ATA capabilities, in accordance with the serial ATA revision 3.0 of Serial ATA International Organization.

SATA consists of an AHCI command layer (with a descriptor-based DMA controller) designed to operate in a system that supports command queuing. It supports FIS-based switching for improved performance in systems that employ port multipliers. Under the command layer, SATA contains the transport and link layers, as outlined in the SATA standard.



**Figure 32-1. SATA controller layer overview**

## 32.2 SATA features summary

The SATA controller includes the following features:

- Complies with the serial ATA 3.0 specification and the AHCI 1.3.1 specification
- Contains a high-speed descriptor-based DMA controller
- Supports the following:
  - Speeds of 1.5 Gb/s (first-generation SATA), 3 Gb/s (second-generation SATA), and 6 Gb/s (third-generation SATA)
  - FIS-based switching
  - Native command queuing (NCQ) commands
  - Port multiplier operation
  - Asynchronous notification
  - SATA Vendor BIST mode

## 32.3 SATA AHCI register descriptions

The SATA controller is compliant to AHCI 1.3.1 programming model. The SATA memory map includes only those registers which have implementation differences from AHCI Specification. For complete registers list, refer to the SATA AHCI Specification.

The registers are divided into global registers and port control registers, as shown in the table below.

**Table 32-1. Register group**

Start address	End address	Description
00h	2Bh	Generic host controller
A0h	FFh	Vendor-specific registers
100h	17Fh	Port 0 port control registers

### 32.3.1 SATA memory map

SATA base address: 320\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	HBA capabilities register (CAP)	32	RO	E537_FF80h
4h	Global HBA control register (GHC)	32	RW	8000_0000h
10h	AHCI version register (VS)	32	RO	0001_0301h
14h	Command completion coalescing control register (CCC_CTL)	32	RW	0001_0110h
24h	HBA capabilities extended register (CAP2)	32	RO	0000_000Ch
A4h	Port config register (PCFG)	32	RW	0032_2002h
A8h	Port Phy1Cfg register (PPCFG)	32	RW	A001_FFFEh
ACh	Port Phy2Cfg register (PP2C)	32	RW	2818_4D1Bh
B0h	Port Phy3Cfg register (PP3C)	32	RW	0E08_1906h
B4h	Port Phy4Cfg register (PP4C)	32	RW	064A_0813h
B8h	Port Phy5Cfg register (PP5C)	32	RW	3FFC_96A4h
BCh	AXI cache control register (AXICC)	32	RW	0010_0010h
C0h	Port AXICfg register (PAXIC)	32	RW	0041_0102h
C8h	Port TransCfg register (PTC)	32	RW	0800_0020h
D0h	Port LinkCfg register (PLC)	32	RW	3800_FF34h
D4h	Port LinkCfg1 register (PLC1)	32	RW	0000_0000h
D8h	Port LinkCfg2 register (PLC2)	32	RW	0000_0000h
E0h	Port LinkStatus1 register (PLS1)	32	RW	0000_0000h
E4h	Port CmdConfig register (PCMDC)	32	RW	0000_0000h
E8h	Port PhyControl status register (PPCS)	32	RW	Table 32-6
F0h	Timer control register (TCR)	32	RW	0000_0100h

Table continues on the next page...

## SATA AHCI register descriptions

Offset	Register	Width (In bits)	Access	Reset value
100h	Port x command list base address register (PxCLB)	32	RW	0000_0000h
104h	Port x command list base address upper 32-bit register (PxCLBU)	32	RW	0000_0000h
108h	Port x FIS base address register (PxFB)	32	RW	0000_0000h
10Ch	Port x FIS base address upper 32-bit register (PxFBU)	32	RW	0000_0000h
110h	Port x interrupt status register (PxIS)	32	RW	0000_0000h
118h	Port x command and status register (PxCMD)	32	RW	0040_0000h
128h	Port x SATA status register (PxSSTS)	32	RO	0000_0000h
12Ch	Port x SATA control register (PxSCTL)	32	RW	0000_0300h
130h	Port x SATA error register (PxSERR)	32	RW	<a href="#">Table 32-6</a>
138h	Port x command issue register (PxCI)	32	RW	0000_0000h
140h	Port x FIS-based switching control register (PxFBS)	32	RW	0000_4000h
170h	Port 0 BIST error register (PBERR)	32	RW	0000_0003h

### 32.3.2 HBA capabilities register (CAP)

#### 32.3.2.1 Offset

Register	Offset
CAP	0h

#### 32.3.2.2 Function

This register indicates basic capabilities of HBA to the driver software.

### 32.3.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	S64A	SNCQ	SSNT F	SMP S	SS S	SAL P	SA L	SCLO	IS S				Reserved	SAM	SPM	FBS S
W																
Reset	1	1	1	0	0	1	0	1	0	0	1	1	0	1	1	1
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PMD	SS C	PS C	NCS					CCCS	EM S	SX S	NP				
W																
Reset	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0

### 32.3.2.4 Fields

Field	Function
31 S64A	Supports 64-bit addressing Indicates whether the HBA can access 64-bit data structures. 0b - The upper 32-bit bits of the port DMA descriptor, the PRD base, and each PRD entry are read-only and treated as 0 by the HBA. 1b - The HBA makes the upper 32-bit bits of the port DMA descriptor, the PRD base, and each PRD entry read/write.
30 SNCQ	Supports native command queuing Indicates whether the HBA supports SATA native command queuing. 0b - The native command queuing is not supported and software should not issue any native command queuing commands. 1b - The HBA handles DMA Setup FISes natively and the auto-activate optimization through that FIS.
29 SSNTF	Supports SNotification register 0b - The HBA does not support the PxSNTF[SNotification] register and its associated functionality. Refer to Section 10.11.1 of the SATA AHCI Specification for more details. Asynchronous notification with a directly attached device is always supported. 1b - The HBA supports the PxSNTF[SNotification] register and its associated functionality.
28 SMPS	Supports mechanical presence switch 0b - This function is not supported. This value is loaded by the BIOS prior to OS initialization. 1b - The HBA supports mechanical presence switches on its ports for use in hot plug operations.
27 SSS	Supports staggered spin-up 0b - This function is not supported. This value is loaded by the BIOS prior to OS initialization. 1b - The HBA supports staggered spin-up on its ports, for use in balancing power spikes.
26 SALP	Supports aggressive link power management 0b - This function is not supported and software treats the PxCMD[ALPE] and PxCMD[ASP] bits as reserved. 1b - The HBA supports auto-generating link requests to the partial or slumber states if there are no commands to process.

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
25 SAL	Supports activity LED 0b - This function is not supported. 1b - The HBA supports a single activity indication output pin. This pin can be connected to an LED on the platform to indicate device activity on any drive.
24 SCLO	Supports command list override 0b - The HBA is not capable of clearing the PxTFD[BSY] and PxTFD[DRQ] bits in the status register to issue a software reset if these bits are still set from a previous operation. 1b - The HBA supports the PxCMD[CLO] bit and its associated function.
23-20 ISS	Interface speed support This bit indicates the maximum speed that the HBA can support on its ports. These encodings match the system software programmable PxSCTL[DET] and PxSCTL[SPD] bits. The settings not shown below are reserved. 0001b - Gen 1 (1.5 Gbps) 0010b - Gen 2 (3 Gbps) 0011b - Gen 3 (6 Gbps)
19 —	Reserved
18 SAM	Supports AHCI mode only The SATA controller may optionally support AHCI access mechanisms only. 0b - In addition to the native AHCI mechanism (through ABAR), the SATA controller implements a legacy, task-file based register interface such as SFF-8038i. 1b - The SATA controller does not implement a legacy, task-file based register interface.
17 SPM	Supports port multiplier This bit indicates whether the HBA supports a port multiplier. 0b - A port multiplier is not supported and a port multiplier may not be attached to this HBA. 1b - A port multiplier using command-based switching is supported and FIS-based switching may be supported.
16 FBSS	FIS-based switching supported This bit must be set if the SPM bit is set. 0b - The HBA does not support FIS-based switching. 1b - The HBA supports port multiplier FIS-based switching.
15 PMD	PIO multiple DRQ block In AHCI 1.2 HBAs, this bit is set to 1. 0b - The HBA only supports single DRQ block data transfers for the PIO command protocol. 1b - The HBA supports multiple DRQ block data transfers for the PIO command protocol.
14 SSC	Slumber state capable This bit indicates whether the HBA supports transitions to the slumber state. 0b - Software must neither allow the HBA to initiate transitions to the slumber state through aggressive link power management nor the PxCMD[ICC] and PxSCTL[IPM] bits in each port must be programmed to disallow device initiated slumber requests. 1b - The HBA and device initiated slumber requests are supported.
13 PSC	Partial state capable This bit indicates whether the HBA can support transitions to the partial state. 0b - Software must neither allow the HBA to initiate transitions to the partial state through aggressive link power management nor the PxCMD[ICC] and PxSCTL[IPM] bits in each port must be programmed to disallow device initiated partial requests. 1b - The HBA and device initiated partial requests are supported.
12-8 NCS	Number of command slots

Table continues on the next page...

Field	Function
	This bit provides value indicating the number of command slots per port supported by this HBA. A minimum of 1 and maximum of 32 slots per port can be supported. The same number of command slots is available on each implemented port.
7 CCCS	Command completion coalescing supported 0b - The HBA does not support command completion coalescing and the CCC_CTL and CCC_PORTS global HBA registers are not implemented. 1b - The HBA supports command completion coalescing as defined in Section 11 of the SATA AHCI Specification. When command completion coalescing is supported, the HBA has implemented the CCC_CTL and the CCC_PORTS global HBA registers.
6 EMS	Enclosure management supported 0b - The HBA does not support enclosure management and the EM_LOC and EM_CTL global HBA registers are not implemented. 1b - The HBA supports enclosure management as defined in Section 12 of the SATA AHCI Specification. When enclosure management is supported, the HBA has implemented the EM_LOC and EM_CTL global HBA registers.
5 SXS	Supports external SATA If this bit is set to 1, software may refer to the PxCMD[ESP] bit to determine whether a specific port has its signal connector externally accessible as a signal only connector (i.e. power is not part of that connector). 0b - The HBA has no SATA port that has a signal only connector externally accessible. 1b - This bit indicates that the HBA has one or more SATA ports that has a signal only connector that is externally accessible (for example, eSATA connector).
4-0 NP	Number of ports This bit provides value indicating the maximum number of ports supported by the HBA silicon. A maximum of 32 ports can be supported. A value of 0h, indicating one port, is the minimum requirement. <b>NOTE:</b> The number of ports indicated in this BIT may be more than the number of ports indicated in the PI register.

### 32.3.3 Global HBA control register (GHC)

#### 32.3.3.1 Offset

Register	Offset
GHC	4h

#### 32.3.3.2 Function

This register controls various global actions of the HBA.

### 32.3.3.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	AE	Reserved															
W																	
Reset	1	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved														MRSRM	E	HR
W															0	0	0
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 32.3.3.4 Fields

Field	Function
31	AHCI enable
AE	This is used by the HBA that supports both legacy mechanisms (such as, SFF-8038i) and AHCI to know when the HBA is running under the AHCI driver. When set, software only communicates with the HBA using AHCI. When cleared, software only communicates with the HBA using legacy mechanisms. Software sets this bit to 1 before accessing other AHCI registers. The implementation of this bit is dependent upon the value of the CAP[SAM] bit. If CAP[SAM] is 0, then GHC[AE] is read-write with a reset value of 0. If CAP[SAM] is 1, then GHC[AE] is read-only with a reset value of 1. 0b - FISes are not posted to memory and no commands are sent through AHCI mechanisms. 1b - Communication to the HBA happens through AHCI mechanisms.
30-3	Reserved
—	
2	MSI revert to single message  This bit is set to 1 only when the following conditions hold: <ul style="list-style-type: none"> <li>• MC[MSIE] = 1 (MSI is enabled)</li> <li>• MC[MMC] &gt; 0 (multiple messages requested)</li> <li>• MC[MME] &gt; 0 (more than one message allocated)</li> <li>• MC[MME] != MC[MMC] (messages allocated not equal to number requested)</li> </ul> When this bit is set to 1, single MSI mode operation is in use and software is responsible for clearing bits in the IS register to clear interrupts. This bit is cleared to '0' by hardware when any of the above four conditions stated is false. This bit is also cleared to 0 when MC[MSIE] = 1 and MC[MME] = 0. In this case, the hardware has been programmed to use single MSI mode, and is not reverting to that mode.  0b - The HBA has not reverted to single MSI mode (i.e. hardware is already in single MSI mode, software has allocated the number of messages requested, or hardware is sharing interrupt vectors if MC[MME] < MC[MMC]). The HBA may revert to single MSI mode when the number of vectors allocated by the host is less than the number requested. 1b - The HBA requested more than one MSI vector but has reverted to using the first vector only.
1	Interrupt enable
IE	This bit enables interrupts from the HBA.

Table continues on the next page...

Field	Function
	0b - All interrupt sources from all ports are disabled (default) 1b - Interrupts are enabled.
0 HR	HBA reset  When set by software, this bit causes an internal reset of the HBA. All state machines that relate to data transfers and queuing return to an idle condition, and all ports are re-initialized through COMRESET (if staggered spin-up is not supported). If staggered spin-up is supported, then it is the responsibility of software to spin-up each port after the reset has completed. When the HBA has performed the reset action, it resets this bit to 0. A software write of 0 has no effect.

### 32.3.4 AHCI version register (VS)

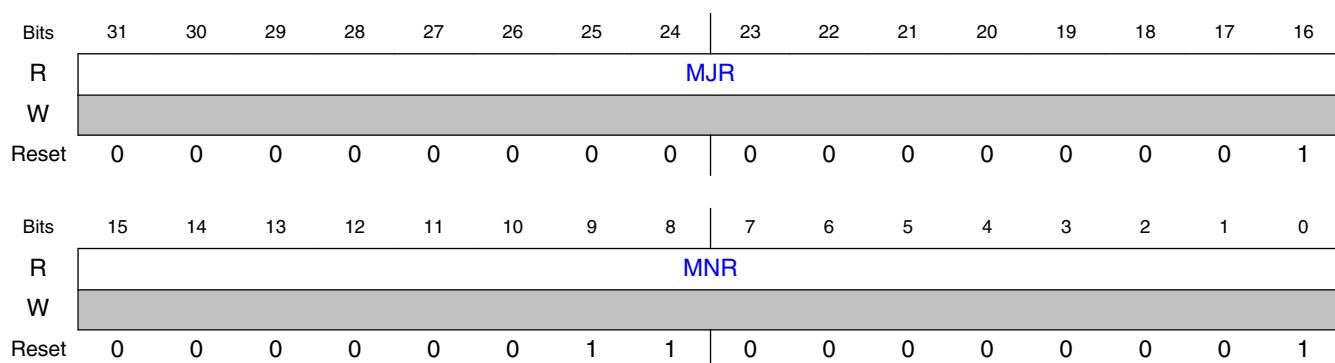
#### 32.3.4.1 Offset

Register	Offset
VS	10h

#### 32.3.4.2 Function

This register indicates the major and minor version of the AHCI specification that the HBA implementation supports. The upper two bytes represent the major version number and the lower two bytes represent the minor version number.

#### 32.3.4.3 Diagram



### 32.3.4.4 Fields

Field	Function
31-16	Major version number
MJR	This bit indicates that the major version is 1.
15-0	Minor version number
MNR	This bit indicates that the minor version is “3.1”.

## 32.3.5 Command completion coalescing control register (CCC\_CTL)

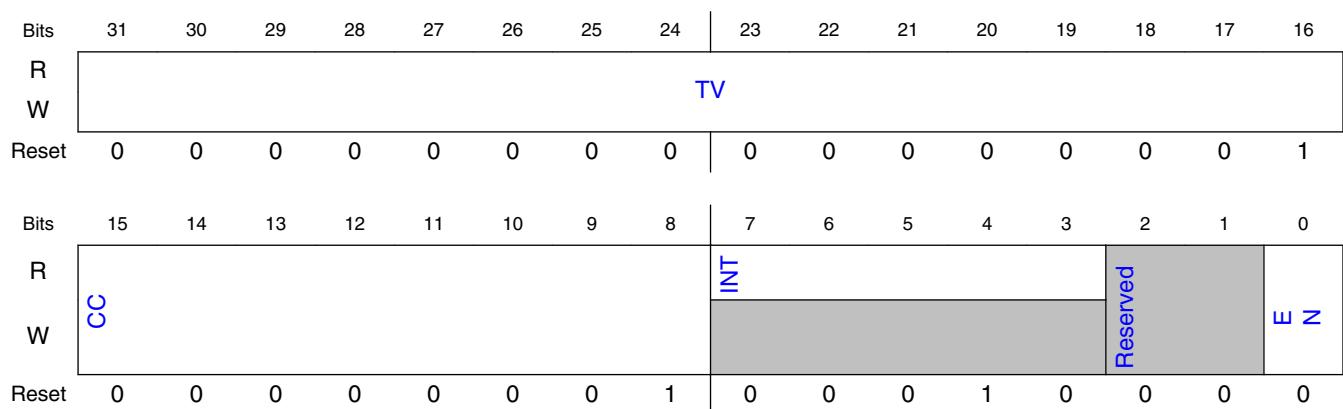
### 32.3.5.1 Offset

Register	Offset
CCC_CTL	14h

### 32.3.5.2 Function

This register configures the command completion coalescing feature for the entire HBA.

### 32.3.5.3 Diagram



### 32.3.5.4 Fields

Field	Function
31-16 TV	Timeout value  The timeout value is specified in 1 ms intervals. The timer accuracy must be within 5%. hCccTimer is loaded with this timeout value. hCccTimer is only decremented when commands are outstanding on selected ports, as defined in Section 11.2 of the SATA AHCI Specification. The HBA signals a CCC interrupt when hCccTimer has decremented to 0. hCccTimer is reset to the timeout value on the assertion of each CCC interrupt. A timeout value of 0 is reserved.
15-8 CC	Command completions  This bit specifies the number of command completions that are necessary to cause a CCC interrupt. The HBA has an internal command completion counter and hCccComplete. hCccComplete is incremented by one each time a selected port has a command completion. When hCccComplete is equal to the command completions value, a CCC interrupt is signaled. The internal command completion counter is reset to 0 on the assertion of each CCC interrupt. A value of 0 for this field disables CCC interrupts being generated based on the number of commands completed, which means CCC interrupts are only generated based on the timer in this case.
7-3 INT	Interrupt  This bit specifies the interrupt used by the CCC feature. This interrupt must be marked as unused in the ports implemented (PI) register by setting the corresponding bit to 0. Therefore, the CCC interrupt corresponds to the interrupt for an unimplemented port on the controller. When a CCC interrupt occurs, the IS[IPS[INT]] bit is asserted to 1. This field also specifies the interrupt vector used for MSI.
2-1 —	Reserved
0 EN	Enable  Software changes only the contents of the TV and CC fields when EN is cleared. When this bit is set, any updated values for the TV and CC fields take effect. 0b - The command completion coalescing feature is disabled and no CCC interrupts are generated. 1b - The command completion coalescing feature is enabled and CCC interrupts may be generated based on timeout or command completion conditions.

### 32.3.6 HBA capabilities extended register (CAP2)

#### 32.3.6.1 Offset

Register	Offset
CAP2	24h

#### 32.3.6.2 Function

This register indicates capabilities of the HBA to driver software.

### 32.3.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												APST	NVMPI	BOH	
W													1	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

### 32.3.6.4 Fields

Field	Function
31-3 —	Reserved
2 APST	Automatic partial to slumber transitions 0b - Automatic partial to slumber transitions are not supported. 1b - The HBA supports automatic partial to slumber transitions.
1 NVMPI	NVMHCl present 0b - The HBA does not support NVMHCl. 1b - The HBA includes support for NVMHCl and the registers at offset 60h-9Fh are valid.
0 BOH	BIOS/OS handoff 0b - The HBA does not support BIOS/OS handoff and the BOHC global HBA register is not implemented. 1b - The HBA supports the BIOS/OS handoff mechanism defined in Section 10.6 of the SATA AHCI Specification and implements the BOHC global HBA register.

### 32.3.7 Port config register (PCFG)

#### 32.3.7.1 Offset

Register	Offset
PCFG	A4h

### 32.3.7.2 Function

This register is used to program and configure the controller port.

### 32.3.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TPSS							
W	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	TPR	S	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PAD						
W	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

### 32.3.7.4 Fields

Field	Function
31-23	Reserved
—	
22-16 TPSS	Microsecond timer post scaler
15	Reserved
—	
14-12 TPRS	Microsecond timer pre scaler
11-9	Reserved
—	
8	Reserved
—	
7-6	Reserved
—	
5-0 PAD	Port address. The settings not defined below are reserved. 000010b - Address configuration/status register set

## 32.3.8 Port Phy1Cfg register (PPCFG)

### 32.3.8.1 Offset

Register	Offset
PPCFG	A8h

### 32.3.8.2 Function

This register controls the configuration of the PHY control layer 1 for port 0. This register holds configuration values for both the OOB generator and OOB detector for each of the two OOB pulses, COMINIT/COMRESET and COMWAKE. The separate values allow multiple configurations of the clocks for running either the OOB generator or detector. The PHY control layer implements the OOB generation logic synchronous TXCLK, which is a 75 MHz input clock to the host controller. The OOB detector logic works synchronous to PMCLK, which is a fixed clock of 150 MHz and used solely for the detector clock.

### 32.3.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESD F	EFS N	PS S	PSS O	ST B	PPNA	PBC E	PBP E	PBP S			FP R	Reserved	SN R	SNM	TTA
W					0	0	0	0	0	0	0	0	0	0	0	1
Reset	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									TTA							
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

### 32.3.8.4 Fields

Field	Function
31 ESDF	Enable signal detect filter  When set, a single de-assertion of signal detect or the coreRxDataValid, when the PHY Init state machine is in the PHYReady state, causes the state machine to exit the PHYReady state and return to a PhyNotReady state. This results in the OOB and speed negotiation running again.
30 ERSN	Enable reset speed negotiation  When set, the PHY control layer enables only a single speed on the RX path during speed negotiation. This speed is determined as the fastest support for the first round falling to the lowest speed for the final round. Each round of speed negotiation is terminated by the host issuing a COMRESET and rerunning OOB before beginning the next round of speed negotiation as detailed in reset speed negotiation (RSN).
29 PSS	PhyControl SerDes slumber  This bit selects SerDes slumber CMU during link slumber. When this bit is set and the controller enters slumber, an extra control signal is applied to the SerDes to slumber the clock block within SerDes. This yields an extra power savings that is SerDes specific.
28 PSSO	PhyControl select SerDes OOB  This bit selects SerDes OOB or internally decoded OOB signalling as input. 0b - Select SerDes decoded OOB signalling 1b - Select internally decoded OOB signalling
27 STB	Status bit  This bit provides the status of the Gen fixed clocks parameter. This bit indicates if the PHY control layer is running from a fixed frequency clock or a variable clock derived from the TX clock of the SerDes.
26 PBPNA	PhyControl BIST pattern no aligns  Setting this bit causes the PHY control pattern generator to transmit each pattern continuously.
25 PBCE	PhyControl BIST clear error  When a pattern mismatch occurs, this bit needs to be set and then negated to clear the error. 0b - Pattern match error bit is not cleared 1b - Clears the pattern match error bit
24 PBPE	PhyControl BIST pattern enable  This bit controls enabling/disabling the PHY control test pattern generation. For more information, refer to <a href="#">PhyControl BIST modes</a> 0b - Disables the PHY control test pattern generation 1b - Enables the PHY control test pattern generation
23-21 PBPS	PhyControl BIST pattern select  The settings not defined below are reserved. 000b - LBP (generator clock only) 001b - LFTP 010b - MFTP 011b - HFTP 100b - PRBS pattern 101b - BIST pattern (default)
20 FPR	Force PHY Ready  This bit determines how PHY Ready is driven. 0b - Normal operation mode 1b - FrcPhyRdy: In this mode, the OOB and speed negotiation states of the PHY Init state machine are bypassed. The PHY Init state machine directly enters PHY Ready after reset. The Tx buffer IDLE control is forced off.

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
19 —	Reserved
18 SNR	Speed negotiation rate When this bit is set to 1, the speed negotiation runs only at the rate programmed in the PxSCTL[SPD] bit.
17 SNM	Speed negotiation method If the SATA host controller supports Gen3 speed, this bit must be set to 1. If the SATA host controller is limited to Gen2 or Gen1 speed, this bit may use the default value 0. 0b - The speed negotiation runs starting at the fastest supported speed down to the Gen1 speed. 1b - The speed negotiation runs starting at the Gen1 speed up to the fastest supported speed.
16-0 TTA	It determines the time period. This bit determines the time period that the controller transmits and waits for ALIGNp during speed negotiation. This value is derived for the PMCLK period.

## 32.3.9 Port Phy2Cfg register (PP2C)

### 32.3.9.1 Offset

Register	Offset
PP2C	ACh

### 32.3.9.2 Function

This register controls the configuration of the PHY control OOB timing for the COMINIT parameters for port 0. For more info on OOB timing setup, refer to [PHY control configuration register OOB timing setup](#).

The default value of this register is overridden by 2818\_4D1F.

**Table 32-2. OOB specifications**

Interval	Specification value (ns)	OOB generator TXCLK (75 MHz)	OOB detector PMCLK (150 MHz)
Negate minimum	$\geq 525.0$	28h	-
Gap nominal	= 320.0	18h	-
Gap maximum	$\leq 525.0$	-	4Dh
Gap minimum <sup>1</sup>	$\geq 200.0$	-	1Fh

1. While calculating the CIBGMN value, add 25 ns to the specification value.

### 32.3.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CINMP								CIBGN							
W																
Reset	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CIBGMX								CIBGMN							
W																
Reset	0	1	0	0	1	1	0	1	0	0	0	1	1	0	1	1

### 32.3.9.4 Fields

Field	Function
31-24	COMINIT negate minimum period
CINMP	OOB function: generator clock
23-16	COMINIT burst gap nominal
CIBGN	OOB function: generator clock
15-8	COMINIT burst gap maximum
CIBGMX	OOB function: detector clock
7-0	COMINIT burst gap minimum
CIBGMN	OOB function: detector clock

## 32.3.10 Port Phy3Cfg register (PP3C)

### 32.3.10.1 Offset

Register	Offset
PP3C	B0h

### 32.3.10.2 Function

This register controls the configuration of the PHY control OOB timing for the COMWAKE parameters for port 0. For more info on OOB timing setup, refer to [PHY control configuration register OOB timing setup](#).

The following table provides the OOB specifications for the chip.

**Table 32-3. OOB specifications**

Interval	Specification value (ns)	OOB generator TXCLK (75 MHz)	OOB detector PMCLK (150 MHz)
Negate minimum	$\geq 175.0$	0Eh	-
Gap nominal	= 106.6	08h	-
Gap maximum <sup>1</sup>	$\leq 175.0$	-	15h
Gap minimum	$\geq 35.0$	-	09h

1. While calculating the CWBGMX value, subtract 25 ns from the specification value.

### 32.3.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CWNMP								CWBGN							
W	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0
Reset	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CWBGMX								CWBGMN							
W	0	0	0	1	1	0	0	1	0	0	0	0	0	1	1	0
Reset	0	0	0	1	1	0	0	1	0	0	0	0	0	1	1	0

### 32.3.10.4 Fields

Field	Function
31-24 CWNMP	COMWAKE negate minimum period OOB function: generator clock
23-16 CWBGN	COMWAKE burst gap nominal OOB function: generator clock
15-8 CWBGMX	COMWAKE burst gap maximum OOB function: detector clock

Table continues on the next page...

Field	Function
7-0	COMWAKE burst gap minimum
CWBGMN	OOB function: detector clock

### 32.3.11 Port Phy4Cfg register (PP4C)

#### 32.3.11.1 Offset

Register	Offset
PP4C	B4h

#### 32.3.11.2 Function

This register controls the configuration of the PHY control burst timing for the COM parameters for port 0. For more info on OOB timing setup, refer to [PHY control configuration register OOB timing setup](#).

The following table provides the OOB specifications for the chip.

**Table 32-4. OOB specifications**

Interval	Specification value (ns)	OOB generator TXCLK (75 MHz)	OOB detector PMCLK (150 MHz)
Signal failure detection	-	-	4Ah
Burst nominal	= 106.6	08h	-
Burst maximum	≤ 109.9 non 113.33	-	0Fh

### 32.3.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PTST								SFD							
W	0	0	0	0	0	1	1	0	0	1	0	0	1	0	1	0
Reset	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BNM								BMX							
W	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	1
Reset	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	1

### 32.3.11.4 Fields

Field	Function
31-24 PTST	Partial to slumber timer  This bit provides a value that specifies the delay that the controller should apply while in partial before entering slumber. The value is based on the system clock divided by 128, total delay = (system clock period) * PTST * 128.
23-16 SFD	Signal failure detection  OOB function: detector clock  If the signal detection de-asserts for a time greater than this, the OOB detector determines this as a line idle and causes the PhyInit state machine to exit the PHY Ready state.
15-8 BNM	COM burst nominal  OOB function: generator clock
7-0 BMX	COM burst maximum  OOB function: detector clock

## 32.3.12 Port Phy5Cfg register (PP5C)

### 32.3.12.1 Offset

Register	Offset
PP5C	B8h

### 32.3.12.2 Function

This register controls the configuration of the PHY control retry interval timing for port 0. For more info on OOB timing setup, refer to [PHY control configuration register OOB timing setup](#).

**Table 32-5. OOB specifications**

Interval	Specification value	Calculated value for PMCLK (150 MHz)
Rate change timer	$\geq 54.2 \mu\text{s} / 4$	03FFh
Retry interval	$\geq 10.0 \text{ ms}$	0C96A8h

### 32.3.12.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R									RCT								
W																RIT	
Reset	0	0	1	1	1	1	1	1		1	1	1	1	1	1	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R									RIT								
W																	
Reset	1	0	0	1	0	1	1	0		1	0	1	0	0	1	0	0

### 32.3.12.4 Fields

Field	Function
31-20 RCT	Rate change timer This bit provides a value based on the $54.2 \mu\text{s}$ for which a SATA device transmits at a fixed rate ALIGNp after OOB has completed. For a fast SerDes, this value is recommended to be $54.2 \mu\text{s} / 4$ . OOB function: detector clock
19-0 RIT	Retry interval timer The calculated value is divided by two and the lower digit of precision is not needed. OOB function: detector clock

## 32.3.13 AXI cache control register (AXICC)

### 32.3.13.1 Offset

Register	Offset
AXICC	BCh

### 32.3.13.2 Function

This register controls the value of the AWCACHE and ARCACHE used to distinguish each address operation on the address bus.

#### AWCACHE/ARCACHE control (Non AHCI standard)

Both AWCACHE and ARCACHE can be controlled during header, command, and status of the data phase AXI bus operations. The values driven onto these signals are controlled by the AXI cache control register for operations related to the header, physical region description, command, and status operations. The values driven onto these signals during a data operation can be further controlled from the relevant PRDT entry as shown in [Table 32-13](#). Note that this control is not an AHCI standard.

The following table summarizes each of the possible attributes that can be set through the cache bits.

**Table 32-6. Attributes list**

ARCACHE[3:0]/AWCACHE[3:0]				Transaction attributes
WA	RA	C	B	
0	0	0	0	Non-cacheable and non-bufferable
0	0	0	1	Bufferable only
0	0	1	0	Cacheable, but do not allocate
0	0	1	1	Cacheable and bufferable, but do not allocate
0	1	0	0	Reserved
0	1	0	1	Reserved
0	1	1	0	Cacheable write-through, allocate on reads only

*Table continues on the next page...*

**Table 32-6. Attributes list (continued)**

ARCACHE[3:0]/AWCACHE[3:0]				Transaction attributes
0	1	1	1	Cacheable write-back, allocate on reads only
1	0	0	0	Reserved
1	0	0	1	Reserved
1	0	1	0	Cacheable write-through, allocate on writes only
1	0	1	1	Cacheable write-back, allocate on writes only
1	1	0	0	Reserved
1	1	0	1	Reserved
1	1	1	0	Cacheable write-through, allocate on both reads and writes
1	1	1	1	Cacheable write-back, allocate on both reads and writes

### 32.3.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				EARC	EAWC	AWCF	AWCD				AWCFD				
W	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ARCP				ARCH				ARCF				ARCA			
W	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

### 32.3.13.4 Fields

Field	Function
31-30	Reserved
—	
29	Enable ARCACHE

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
EARC	This bit provides the control from the PRDT entries used during the data phase of this operation (global control).
28	Enable AWCACHE
EAWC	This bit provides the control from the PRDT entries used during the data phase of this operation (global control).
27-24	Address write cache FIS
AWCF	This bit provides the value driven onto AWCACHE when the AXI master is posting a status FIS write address to the memory controller.
23-20	Address write cache data
AWCD	This bit provides the value driven onto AWCACHE when the AXI master is posting a data burst write address to the memory controller when the data burst is not the final burst in the transfer.
19-16	Address write cache final data
AWCFD	This bit provides the value driven onto AWCACHE when the AXI master is posting a data burst write address to the memory controller when the data burst is the final burst in the transfer.
15-12	Address read cache PRD
ARCP	This bit provides the value driven onto ARCACHE when the AXI master is posting a PRD read address to the memory controller.
11-8	Address read cache header
ARCH	This bit provides the value driven onto ARCACHE when the AXI master is posting a header or physical region descriptor read address to the memory controller.
7-4	Address read cache FIS
ARCF	This bit provides the value driven onto ARCACHE when the AXI master is posting a command FIS read address to the memory controller.
3-0	Address read cache ATAPI
ARCA	This bit provides the value driven onto ARCACHE when the AXI master is posting a data burst.

### 32.3.14 Port AXICfg register (PAXIC)

#### 32.3.14.1 Offset

Register	Offset
PAXIC	C0h

#### 32.3.14.2 Function

This register controls the configuration of the AXI bus operation for port 0. The AXIPE and ADBW bits can only be configured through access to port 0 configuration space.

### 32.3.14.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16	
R	Reserved				ENP E	Reserved			AAO	ECM	OTL				MARIDD			
W	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0	
R	MARID				MAWIDD				MAWID				Reserved				Reserved	
W	0	0	0	0	0	0	0	1	Reserved				0	0	0	0	1	0
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0

### 32.3.14.4 Fields

Field	Function
31-29	Reserved
—	
28	Enable non-zero 4 MB PRD entries
ENPE	
27-26	Reserved
—	
25	Allow address overwrite
AAO	This bit allows the PxCLB[9:0] and PxFB[7:0] bits to be overwritten.
24	Enable the context management
ECM	This bit enables the context management system in the memory.
23-20	Outstanding transfer limit
OTL	<p>This bit limits the maximum number of outstanding transfers supported.</p> <ul style="list-style-type: none"> <li>For the 264 DWord transport layer implementation, this can be programmed between 1 and 16</li> <li>For the 136 DWord transport layer implementation, this can be programmed between 1 and 8</li> <li>For the 72 DWord transport layer implementation, this can be programmed between 1 and 4</li> </ul> <p>The settings not defined below are reserved.</p> <p>0000b - Transfers limited by the transport layer FIFO space/fill level.</p>
19-16	Memory address read ID
MARIDD	This bit provides memory address read ID for data transfers.
15-12	Memory address read ID
MARID	This bit provides memory address read ID for non data transfers.

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
11-8 MAWIDD	Memory address write ID This bit provides memory address write ID for data transfers.
7-4 MAWID	Memory address write ID This bit provides memory address write ID for non data transfers.
3-2 —	Reserved
1-0 —	Reserved

## 32.3.15 Port TransCfg register (PTC)

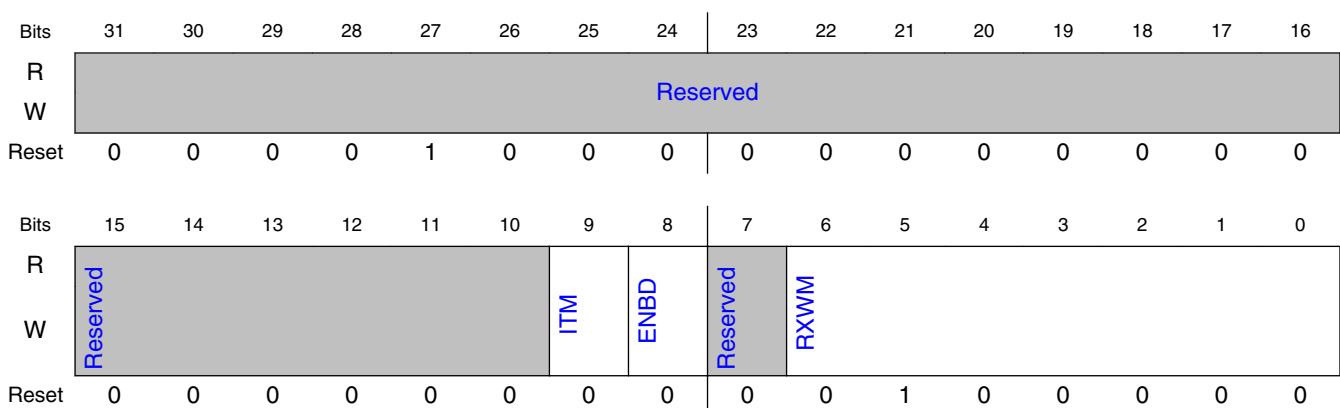
### 32.3.15.1 Offset

Register	Offset
PTC	C8h

### 32.3.15.2 Function

This register controls the configuration of the transport layer for port 0.

### 32.3.15.3 Diagram



### 32.3.15.4 Fields

Field	Function
31-10 —	Reserved
9 ITM	Initialize transport memories 0b - The machine runs writing 0 to all locations within the memory 1b - Causes the memory initialize hardware state machine to enter reset
8 ENBD	Enable back down When a port multiplier is attached and an attempt to send a command to an attached drive results in three consecutive retries due to R_ERR receptions, the command removes for the transport layer and returns to the pending queue. This is to avoid unnecessary retries owing to the device trying to send a set device bits FIS while the host is trying to send a command. This feature improves performance as it allows the controller to queue any pending commands to the other drives.
7 —	Reserved
6-0 RXWM	RxWaterMark This bit sets the minimum number of free location within the RX FIFO before the watermark is exceeded. The transport layer in turn instructs the link layer to transmit HOLDS to the transmitting end.  <b>NOTE:</b> It can take some time for the HOLDS to get to the other end and in the interim period there must be enough room in the FIFO to absorb all data that could arrive. An initial value of 7'h29 is recommended.

### 32.3.16 Port LinkCfg register (PLC)

#### 32.3.16.1 Offset

Register	Offset
PLC	D0h

#### 32.3.16.2 Function

This register controls the configuration of the link layer for port 0.

### 32.3.16.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R W	PMPRA							POE	PRT								
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R W	AIR								EPNR	S4A	RXSE	TXSE	TXPJ	TXC	RXBC	TXBC	
Reset	1	1	1	1	1	1	1	1	0	0	1	1	0	1	0	0	0

### 32.3.16.4 Fields

Field	Function
31-27 PMPRA	Power management primitive rate acknowledge  This bit determines the number of PMACK primitives sent when a power management state transition is requested by the host.
26 POE	Primitive override enable  When set, this bit enables the replacement of a single primitive, as specified by override primitive/CD, when the link layer state machine is in the Prim override state. This bit must be set to enable this feature.
25-16 PRT	PHY Ready timer  This bit specifies the timeout value of the PHY Ready timer. If EnPhyReadyTimeOut is set, the link layer counts down on every rising edge of TX clock, as long as PHY Ready is de-asserted. When the counter reaches zero, a PhyReset is issued to the PHY to try and re-establish communications with the far-end. The timer is initially loaded with a value equal to the concatenation of { PHY Ready Timer, 9'h000}.
15-8 AIR	ALIGN insertion rate  The SATA AHCI Specification requires that the link layer sends a pair of ALIGN primitives at least every 254 DWords of data. This is achieved by setting ALIGN insertion rate to "11111111". However, for test purposes it is possible to send ALIGNs at a higher rate. This can be achieved by setting ALIGN insertion rate to a lower value i.e. (ALIGN insertion rate-1). DWords are sent by the link layer between each set of ALIGN primitive pairs.  <b>NOTE:</b> If the S4A bit is set, the ALIGN insertion rate should not be set to four or less. If S4A is not set, the ALIGN insertion rate should not be set to two or less.
7 EPNRT	Enable PHY not ready timer  If PHY Ready is de-asserted for a long time as specified by PRT bit, then this bit, when asserted, enables the link layer to re-issue a PhyReset, thereby re-initiating OOB.
6 S4A	Send 4 Aligns  If this bit is asserted, four ALIGN primitives are transmitted at the specified rate, instead of the normal two ALIGN primitives.
5 RXSE	Rx scramble enable

Table continues on the next page...

Field	Function
	If this bit is asserted, the de-scrambling of the receive data is enabled as per the SATA AHCI Specification.
4 TXSE	Tx scramble enable  If this bit is asserted, the scrambling of the transmit data is enabled as per the SATA AHCI Specification.
3 TXPJ	Tx Prim junk  If this bit is de-asserted, the scrambled junk data is sent after a CONT primitive, as per the SATA AHCI Specification. If this bit is asserted, then the single character 32'hDEADBEEF is sent continuously to aid debug.
2 TXC	Tx CONT  If this bit is asserted, the transmission of CONT primitives is enabled. If de-asserted, then long sequences of repeated primitives can be sent by the link layer.
1 RXBC	Rx bad CRC  When a rising edge is detected on this bit, it causes a bad CRC to be detected for the current frame. This bit has must be set to enable this feature.
0 TXBC	Tx bad CRC  A bad CRC (inverted value of the correct CRC) value is transmitted for one FIS only by the link layer when a rising edge is detected on this signal. This bit must be set to enable this feature.

### 32.3.17 Port LinkCfg1 register (PLC1)

#### 32.3.17.1 Offset

Register	Offset
PLC1	D4h

#### 32.3.17.2 Function

This register controls the configuration of the link layer for port 0.

### 32.3.17.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved									CD	POS						
W										0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 32.3.17.4 Fields

Field	Function
31-7	Reserved
—	
6 CD	Data character or primitive  This bit specifies whether the data used during the primitive override should be a data character or a primitive. For example, if CD = 1, Prim override state = L_SendEOF and override primitive = WTRM, then a WTRM primitive is inserted into the data stream instead of an EOF (whenever a rising edge is seen on PLC[POE]. If CD = 0, then a normal data character (as specified by PLC2[OP]) is inserted into the data stream instead of the EOF.
5-0 POS	Primitive override state  This bit is used in the primitive override debug functionality. When the link layer detects a positive edge on PLC[POE], it overrides the next primitive that would be inserted during the PLC1[POS], with the data specified by the PLC2[OP] and PLC1[CD] configuration bits.

### 32.3.18 Port LinkCfg2 register (PLC2)

#### 32.3.18.1 Offset

Register	Offset
PLC2	D8h

### 32.3.18.2 Function

This register controls the configuration of the link layer for port 0.

### 32.3.18.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	OP																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	OP																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 32.3.18.4 Fields

Field	Function
31-0	Override primitive
OP	This bit specifies the data to be used in the primitive override debug functionality that is described in the definition of <a href="#">Port LinkCfg1 register (PLC1)</a> .

## 32.3.19 Port LinkStatus1 register (PLS1)

### 32.3.19.1 Offset

Register	Offset
PLS1	E0h

### 32.3.19.2 Function

This register indicates the status of the link layer for port 0. This register acts as an accumulator for the SerDes errors. Each counter can be cleared by writing 8'hFF to the individual byte.

### 32.3.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	KCEC								PIEC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CEC								DEC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 32.3.19.4 Fields

Field	Function
31-24 KCEC	Kchar error count  This bit provides the number of DWords that have been received from the PHY, where one or more control character errors have been detected. A value of 255 indicates an error count of 255 or more as this counter does not wrap around to zero. The count value is updated with its current value each time the LinkStatus1 register is read.
23-16 PIEC	PHY internal error count  This bit provides the number of DWords that have been received from the PHY, where one or more internal errors have been detected. A value of 255 indicates an error count of 255 or more as this counter does not wrap around to zero. The count value is updated with its current value each time the LinkStatus1 register is read.
15-8 CEC	Code error count  This bit provides the number of DWords that have been received from the PHY, where one or more code errors have been detected. A value of 255 indicates an error count of 255 or more as this counter does not wrap around to zero. The count value is updated with its current value each time the LinkStatus1 register is read.
7-0 DEC	Disparity error count  This bit provides the number of DWords that have been received from the PHY, where one or more disparity errors have been detected. A value of 255 indicates an error count of 255 or more as this counter does not wrap around to zero. The count value is updated with its current value each time the LinkStatus1 register is read.

## 32.3.20 Port CmdConfig register (PCMDC)

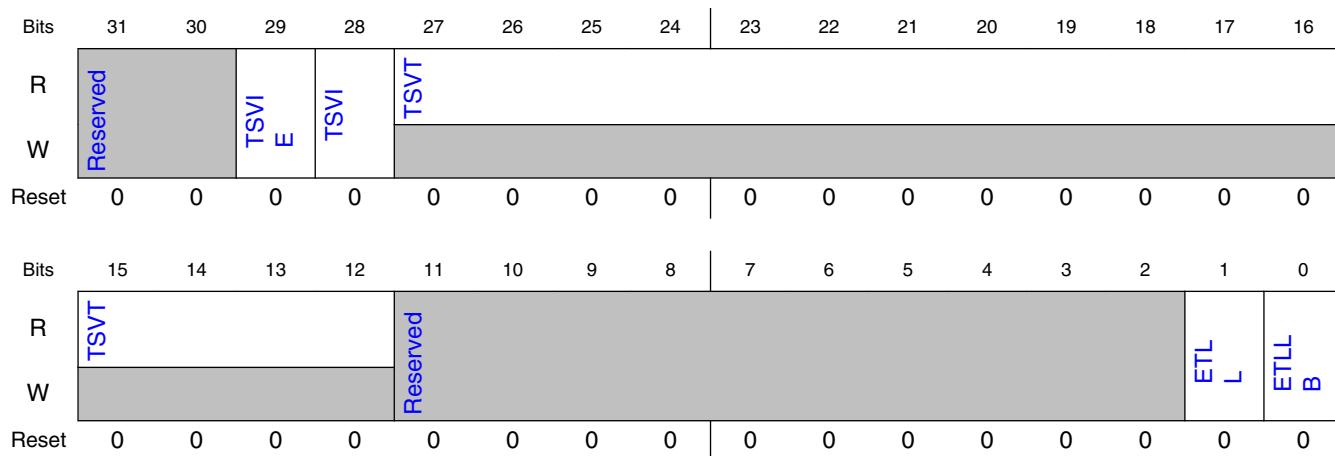
### 32.3.20.1 Offset

Register	Offset
PCMDC	E4h

### 32.3.20.2 Function

This register controls the operation of the command layer for port 0.

### 32.3.20.3 Diagram



### 32.3.20.4 Fields

Field	Function
31-30	Reserved
—	
29 TSVIE	Trustzone slave ID violation interrupt enable The interrupt is seen on bit 3 of the interrupt status register in the general register group. Refer to the SATA AHCI Specification for more details.
28	Trustzone slave ID violation interrupt

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
TSVI	
27-12	Trustzone Slave ID of violating transaction
TSVT	When the slave port rejects a read or write as slave error due to security violation, this register records the AXI ID of the violating transaction.
11-2	Reserved
—	
1	Enable transport layer loopback
ETLL	
0	Enable transport layer loopback in the BIST-L mode
ETLLB	This bit must be set to 1 before performing the BIST-L test.

### 32.3.21 Port PhyControl status register (PPCS)

#### 32.3.21.1 Offset

Register	Offset
PPCS	E8h

#### 32.3.21.2 Function

This register controls the operation of the status of the phy control layer for port 0.

### 32.3.21.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	PHYCE		PHYDE		PHYKC		PHYD										
W																	
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	PHYD				CCAC		CCA						PCTRLS				
W																	
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

### 32.3.21.4 Fields

Field	Function
31-30	Current 2-bit code error
PHYCE	This bit provides snapshot within the PHY control layer.
29-28	Current 2-bit disparity error
PHYDE	This bit provides snapshot within the PHY control layer.
27	Current 1-bit K character
PHYKC	This bit provides snapshot within the PHY control layer.
26-11	Current 16-bit data
PHYD	This bit provides snapshot within the PHY control layer.
10	Comma alignment has changed
CCAC	
9-5	Current comma alignment
CCA	
4-0	PHY control state
PCTRLS	

### 32.3.22 Timer control register (TCR)

### 32.3.22.1 Offset

Register	Offset
TCR	F0h

### 32.3.22.2 Function

This register controls the operation of the timer prescaler that configures a 10 µs pulse generator used to control the operation of the slumber timer. This pulse generator is used in port 0.

### 32.3.22.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved				TPS												
W																	
Reset	0	0	0	0	0	0	0	1		0	0	0	0	0	0	0	0

### 32.3.22.4 Fields

Field	Function
31-13	Reserved
—	
12-0	Timer preScalar value
TPS	The system clock is divided by the ratio to generate a 10 µs clock pulse to run the port layers.

### 32.3.23 Port x command list base address register (PxCLB)

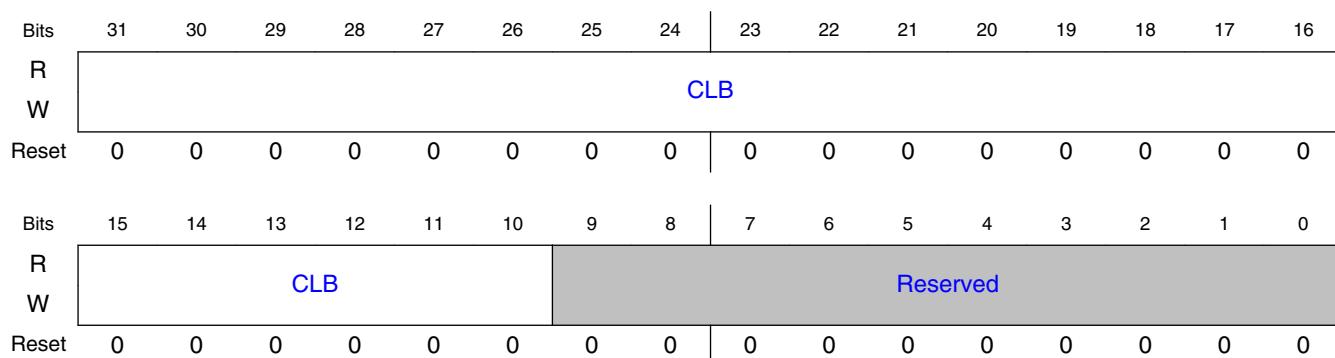
### 32.3.23.1 Offset

Register	Offset
PxCLB	100h

### 32.3.23.2 Function

This bit indicates the 32-bit base physical address for the command list for this port.

### 32.3.23.3 Diagram



### 32.3.23.4 Fields

Field	Function
31-10 CLB	Command list base address This bit indicates the 32-bit base physical address for the command list for this port. This base is used when fetching commands to execute. The structure pointed to by this address range is 1 KB in length. This address must be 1 KB aligned as indicated by bits [9:0] being read only.
9-0 —	Reserved

### 32.3.24 Port x command list base address upper 32-bit register (PxCLBU)

### 32.3.24.1 Offset

Register	Offset
PxCLBU	104h

### 32.3.24.2 Function

Indicates the upper 32-bits for the command list base physical address for this port. This base is used when fetching commands to execute. This register shall be read only '0' for HBAs that do not support 64-bit addressing.

### 32.3.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CLBU															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLBU															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 32.3.24.4 Fields

Field	Function
31-0	Command list base address upper
CLBU	This bit indicates the upper 32 bits for the command list base physical address for this port. This base is used when fetching commands to execute. This bit is read only '0' for HBAs that do not support 64-bit addressing.

### 32.3.25 Port x FIS base address register (PxFB)

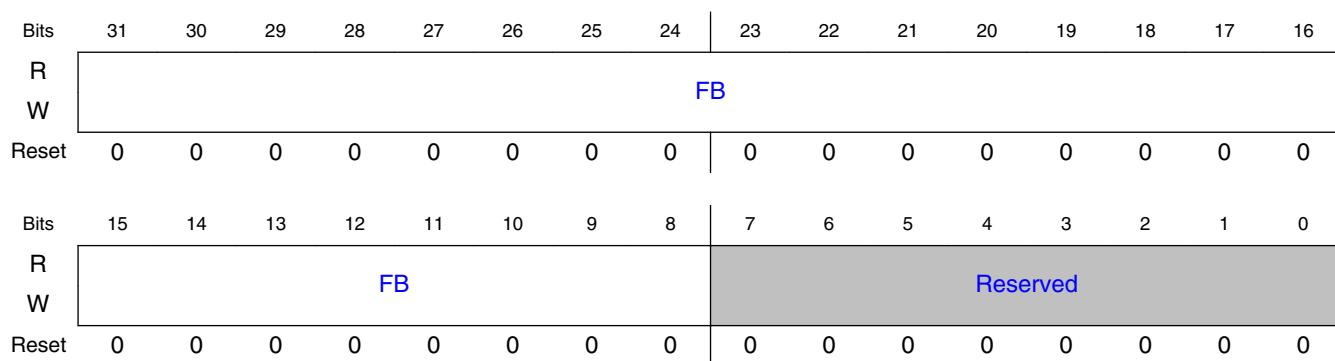
### 32.3.25.1 Offset

Register	Offset
PxFB	108h

### 32.3.25.2 Function

Indicates the 32-bit base physical address for received FISes. The structure pointed to by this address range is 256 bytes in length. This address must be 256-byte aligned as indicated by bits 07:00 being read only.

### 32.3.25.3 Diagram



### 32.3.25.4 Fields

Field	Function
31-8	FIS base address
FB	This bit indicates the 32-bit base physical address for received FISes. The structure pointed to by this address range is 256 bytes in length. This address must be 256 bytes aligned as indicated by bits [7:0] being read only. When FIS-based switching is in use, this structure is 4 KB in length and the address is 4 KB aligned.
7-0	Reserved
—	

## 32.3.26 Port x FIS base address upper 32-bit register (PxFBU)

### 32.3.26.1 Offset

Register	Offset
PxFBU	10Ch

### 32.3.26.2 Function

Indicates the upper 32-bits for the received FIS base physical address for this port. This register shall be read only '0' for HBAs that do not support 64-bit addressing.

### 32.3.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									FBU							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									FBU							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 32.3.26.4 Fields

Field	Function
31-0	FIS base address upper
FBU	This bit indicates the upper 32 bits for the received FIS base physical address for this port. This register is read only '0' for HBAs that do not support 64-bit addressing.

## 32.3.27 Port x interrupt status register (PxIS)

### 32.3.27.1 Offset

Register	Offset
PxIS	110h

### 32.3.27.2 Function

Port x interrupt status register

### 32.3.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	CPDS	TFE S	HBF S	HBDS	IF S	NTF S	Reserved	OF S	IPMS	PRC S	Reserved						
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	Reserved								DMPS	PC S		DP S	UF S	SDB S	DS S	PS S	DHR S
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 32.3.27.4 Fields

Field	Function
31 CPDS	Cold port detect status When set, a device status has changed as detected by the cold presence detect logic. This bit can either be set due to a non-connected port receiving a device or a connected port having its device removed. This bit is only valid if the port supports cold presence detect as indicated by PxCMD[CPD] set to 1.
30 TFES	Task file error status This bit is set whenever the status register is updated by the device and the error bit (bit 0 of the Status field in the received FIS) is set.
29 HBFS	Host bus fatal error status This bit indicates that the HBA encountered a host bus error that it cannot recover from, such as a bad software pointer. In PCI, such an indication would be a target or master abort.
28	Host bus data error status

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
HBDS	This bit indicates that the HBA encountered a data error (uncorrectable ECC/parity) when reading from or writing to system memory.
27	Interface fatal error status
IFS	This bit indicates that the HBA encountered an error on the SATA interface which caused the transfer to stop. Refer to Section 6.1.2 of the SATA AHCI Specification for more details.
26	Interface non-fatal error status
NTFS	This bit indicates that the HBA encountered an error on the SATA interface but was able to continue operation. Refer to Section 6.1.2 of the SATA AHCI Specification for more details.
25	Reserved
—	
24	Overflow status
OFS	This bit indicates that the HBA received more bytes from a device than was specified in the PRD table for the command.
23	Incorrect port multiplier status
IPMS	This bit indicates that the HBA received a FIS from a device that did not have a command outstanding. This bit is set during enumeration of devices on a port multiplier due to the normal port multiplier enumeration process. It is recommended that IPMS only be used after enumeration is complete on the port multiplier. IPMS is not set when an asynchronous notification is received (a set device bits FIS with the notification 'N' bit set to 1).
22	PhyRdy change status
PRCS	When set, this bit indicates the internal PhyRdy signal changed state. This bit reflects the state of PxSERR[DIAG[N]]. To clear this bit, software must clear PxSERR[DIAG[N]] to 0.
21-8	Reserved
—	
7	Device mechanical presence status
DMPS	When set, this bit indicates that a mechanical presence switch associated with this port has been opened or closed, which may lead to a change in the connection state of the device. This bit is only valid if both CAP[SMPS] and PxCMD[MPSP] are set to 1.
6	Port connect change status
PCS	This bit reflects the state of PxSERR[DIAG[X]]. This bit is only cleared when PxSERR[DIAG[X]] is cleared. 0b - No change in current connect status 1b - Change in current connect status
5	Descriptor processed
DPS	A PRD with the 'l' bit set has transferred all of its data. Refer to Section 5.4.2 of the SATA AHCI Specification for more details.
4	Unknown FIS interrupt
UFS	When set, this bit indicates that an unknown FIS was received and has been copied into system memory. This bit is cleared to 0 by software clearing the PxSERR[DIAG[F]] bit to 0.  <b>NOTE:</b> This bit does not directly reflect the PxSERR[DIAG[F]] bit. PxSERR[DIAG[F]] is set immediately when an unknown FIS is detected, whereas this bit is set when that FIS is posted to memory. Software should wait to act on an unknown FIS until this bit is set to 1 or the two bits may become out of sync.
3	Set device bits interrupt
SDBS	A set device bits FIS has been received with the 'l' bit set and has been copied into system memory.

Table continues on the next page...

Field	Function
2 DSS	DMA setup FIS interrupt A DMA setup FIS has been received with the 'I' bit set and has been copied into system memory.
1 PSS	PIO setup FIS interrupt A PIO setup FIS has been received with the 'I' bit set, it has been copied into system memory, and the data related to that FIS has been transferred. This bit is set even if the data transfer resulted in an error.
0 DHRS	Device to host register FIS interrupt A D2H register FIS has been received with the 'I' bit set, and has been copied into system memory.

### 32.3.28 Port x command and status register (PxCMD)

#### 32.3.28.1 Offset

Register	Offset
PxCMD	118h

#### 32.3.28.2 Function

When set to '1', indicates that this port supports Port Multiplier FIS-based switching.  
When cleared to '0', indicates that this port does not support FIS-based switching.

#### 32.3.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICC				AS P	ALP E	DLAE	ATAPI	APST E	FBSC P	ES P	CPD	MPS P	HPCP	PMA	CP S
W					0	0	0	0	0	1	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CR	F R	MPS S	CCS					Reserved			FR E	CLO	POD	SUD	S T
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 32.3.28.4 Fields

Field	Function
31-28 ICC	<p>Interface communication control</p> <p>This bit is used to control power management states of the interface. If the Link layer is currently in the L_IDLE state, writes to this field causes the HBA to initiate a transition to the interface power management state requested. If the Link layer is not currently in the L_IDLE state, writes to this field has no effect. When system software writes a non-reserved value other than No-Op (0h), the HBA performs the action and update this bit back to Idle (0h). If software writes to this bit to change the state to a state the link is already in (i.e. interface is in the active state and a request is made to go to the active state), the HBA takes no action and returns this bit to Idle. If the interface is in a low power state and software wants to transition to a different low power state, software must first bring the link to active and then initiate the transition to the desired low power state.</p> <p>The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - No-Op/Idle: When software reads this value, it indicates the HBA is ready to accept a new interface control command, although the transition to the previously selected state may not yet have occurred.</li> <li>0001b - Active: Causes the HBA to request a transition of the interface into the active state.</li> <li>0010b - Partial: Causes the HBA to request a transition of the interface to the partial state. The SATA device may reject the request and the interface remains in its current state.</li> <li>0110b - Slumber: Causes the HBA to request a transition of the interface to the slumber state. The SATA device may reject the request and the interface must remain in its current state.</li> </ul>
27 ASP	<p>Aggressive slumber/partial</p> <p>If CAP[SALP] is cleared to 0, software treats this bit as reserved. Refer to Section 8.3.1.3 of the SATA AHCI Specification for more details.</p> <ul style="list-style-type: none"> <li>0b - When ALPE is set, the HBA aggressively enters the partial state when it clears the PxCl register and the PxSACT register is cleared or when it clears the PxSACT register and the PxCl register is cleared.</li> <li>1b - When ALPE is set, the HBA aggressively enters the slumber state when it clears the PxCl register and the PxSACT register is cleared or when it clears the PxSACT register and the PxCl register is cleared.</li> </ul>
26 ALPE	<p>Aggressive link power management enable</p> <p>Software only sets this bit to 1 if CAP[SALP] is set to 1. If CAP[SALP] is cleared to 0, software treats this bit as reserved. Refer to Section 8.3.1.3 of the SATA AHCI Specification for more details.</p> <ul style="list-style-type: none"> <li>0b - The aggressive link power management is disabled</li> <li>1b - The HBA aggressively enters a lower link power state (partial or slumber) based upon the setting of the ASP bit.</li> </ul>
25 DLAE	<p>Drive LED on ATAPI enable</p> <p>Refer to Section 10.11 of the SATA AHCI Specification for more details.</p> <p><b>NOTE:</b> The controller does not support ATAPI functionality.</p> <ul style="list-style-type: none"> <li>0b - The HBA only drives the LED pin active for commands if PxCMD[ATAPI] is set to 0.</li> <li>1b - The HBA drives the LED pin active for commands regardless of the state of PxCMD[ATAPI].</li> </ul>
24 ATAPI	<p>Device is ATAPI</p> <p>This bit is used by the HBA to control whether or not to generate the desktop LED when commands are active. Refer to Section 10.11 of the SATA AHCI Specification for more details.</p> <p><b>NOTE:</b> The controller does not support ATAPI functionality.</p> <ul style="list-style-type: none"> <li>0b - The connected device is not an ATAPI device</li> <li>1b - The connected device is an ATAPI device.</li> </ul>

Table continues on the next page...

Field	Function
23 APSTE	Automatic partial to slumber transitions enabled Software only sets this bit to 1 if CAP2[APST] is set to 1. If CAP2[APST] is cleared to 0, software treats this bit as reserved. 0b - The HBA does not perform automatic partial to slumber transitions. 1b - The HBA performs automatic partial to slumber transitions.
22 FBSCP	FIS-based switching capable port This bit may only be set to 1 if both CAP[SPM] and CAP[FBSS] are set to 1. 0b - This port does not support FIS-based switching. 1b - This port supports port multiplier FIS-based switching.
21 ESP	External SATA port When this bit is set to 1, the CAP[SXS] bit is set to 1. ESP is mutually exclusive with the HPCP bit in this register. If ESP is set to 1, then the port may experience hot plug events. 0b - This port's signal connector is not externally accessible on a signal only connector. 1b - This port's signal connector is externally accessible on a signal only connector (e.g. eSATA connector).
20 CPD	Cold presence detection When this bit is set to 1, PxCMD[HPCP] must also be set to 1. 0b - Platform does not support cold presence detection on this port. 1b - Platform supports cold presence detection on this port.
19 MPSP	Mechanical presence switch attached to port When this bit is set to 1, PxCMD[HPCP] should also be set to 1. 0b - The platform does not support a mechanical presence switch attached to this port. 1b - The platform supports the mechanical presence switch attached to this port.
18 HPCP	Hot plug capable port HPCP is mutually exclusive with the ESP bit in this register. 0b - Indicates that this port's signal and power connectors are not externally accessible through a joint signal and power connector. 1b - Indicates that this port's signal and power connectors are externally accessible through a joint signal and power connector for blindmate device hot plug.
17 PMA	Port multiplier attached This bit is read/write for HBAs that support a port multiplier (CAP[SPM] = 1). This bit is read-only for HBAs that do not support a port multiplier (CAP[SPM] = 0). Software is responsible for detecting whether a port multiplier is present; hardware does not auto-detect the presence of a port Multiplier. Software only sets this bit to 1 when PxCMD[ST] is cleared to 0. 0b - A port multiplier is not attached to the HBA for this port. 1b - A port multiplier is attached to the HBA for this port.
16 CPS	Cold presence state The CPS bit reports whether a device is currently detected on this port through cold presence detection. 0b - The HBA detects through cold presence that there is no device attached to this port. 1b - The HBA detects through cold presence that a device is attached to this port.
15 CR	Command list running When set, the command list DMA engine for the port is running. Refer to AHCI state machine in Section 5.3.2 of the SATA AHCI Specification for more details.
14 FR	FIS receive running When set, the FIS receive DMA engine for the port is running. Refer to Section 10.3.2 of the SATA AHCI Specification for more details.
13 MPSS	Mechanical presence switch state

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
	This bit reports the state of a mechanical presence switch attached to this port. If CAP[SMPS] is set to 0, this bit is cleared to 0. Software uses this bit only if both CAP[SMPS] and PxCMD[MPSP] are set to 1. 0b - If CAP[SMPS] is set to 1, the mechanical presence switch is closed. 1b - If CAP[SMPS] is set to 1, the mechanical presence switch is open.
12-8 CCS	Current command slot  This bit is valid when PxCMD[ST] is set to 1 and is set to the command slot value of the command that is currently being issued by the HBA. When PxCMD[ST] transitions from 1 to 0, this bit resets to 0. After PxCMD[ST] transitions from 0 to 1, the highest priority slot to issue from next is command slot 0. After the first command has been issued, the highest priority slot to issue from next is PxCMD[CCS] + 1. For example, after the HBA has issued its first command, if CCS = 0h and PxCl is set to 3h, the next command that will be issued is from command slot 1.
7-5 —	Reserved
4 FRE	FIS receive enable  System software must not set this bit until PxFB (and PxFBU) have been programmed with a valid pointer to the FIS receive area. If software wishes to move the base, this bit must first be cleared, and software must wait for the FR bit in this register to be cleared. Refer to Section 10.3.2 of the SATA AHCI Specification for more details. 0b - Received FISes are not accepted by the HBA, except for the first D2H register FIS after the initialization sequence, and no FISes are posted to the FIS receive area. 1b - The HBA posts received FISes into the FIS receive area pointed to by PxFB (and for 64-bit HBAs, PxFBU).
3 CLO	Command list override  This bit is only set to 1 immediately prior to setting the PxCMD[ST] bit to 1 from a previous value of 0. Setting this bit to 1 at any other time is not supported and results in indeterminate behavior. Software must wait for CLO to be cleared to 0 before setting PxCMD[ST] to 1. 0b - The HBA sets this bit to 0 when PxTFD[STS[BSY]] and PxTFD[STS[DRQ]] have been cleared to 0. A write to this register with a value of 0 has no effect. 1b - Causes PxTFD[STS[BSY]] and PxTFD[STS[DRQ]] to be cleared to 0. This allows a software reset to be transmitted to the device regardless of whether the BSY and DRQ bits are still set in the PxTFD[STS] register. The HBA sets this bit to 0 when PxTFD[STS[BSY]] and PxTFD[STS[DRQ]] have been cleared to 0. A write to this register with a value of 0 has no effect. This bit is only set to 1 immediately prior to setting the PxCMD[ST] bit to 1 from a previous value of 0. Setting this bit to 1 at any other time is not supported and results in indeterminate behavior. Software must wait for CLO to be cleared to 0 before setting PxCMD[ST] to 1.
2 POD	Power-on device  This bit is read/write for HBAs that support cold presence detection on this port as indicated by PxCMD[CPD] set to 1. This bit is read only '1' for HBAs that do not support cold presence detect. When set, the HBA sets the state of a pin on the HBA to 1 so that it may be used to provide power to a cold-presence detectable port.
1 SUD	Spin-up device  This bit is read/write for HBAs that support staggered spin-up through CAP[SSS]. This bit is read only '1' for HBAs that do not support staggered spin-up. On an edge detect from 0 to 1, the HBA starts a COMRESET initialization sequence to the device. Clearing this bit to 0 does not cause any OOB signal to be sent on the interface. When this bit is cleared to 0 and PxSCTL[DET] = 0h, the HBA enters listen mode as detailed in Section 10.10.1 of the SATA AHCI Specification.
0 ST	Start  When this bit is changed from 0 to 1, the HBA starts processing the command list at entry 0. When this bit is changed from 1 to 0, the PxCl register is cleared by the HBA upon the HBA putting the controller into an idle state. This bit is only set to 1 by software after PxCMD[FRE] has been set to 1. Refer to Section 10.3.1 of the SATA AHCI Specification for more details.

Field	Function
	0b - The HBA does not process the command list. 1b - The HBA processes the command list.

### 32.3.29 Port x SATA status register (PxSSTS)

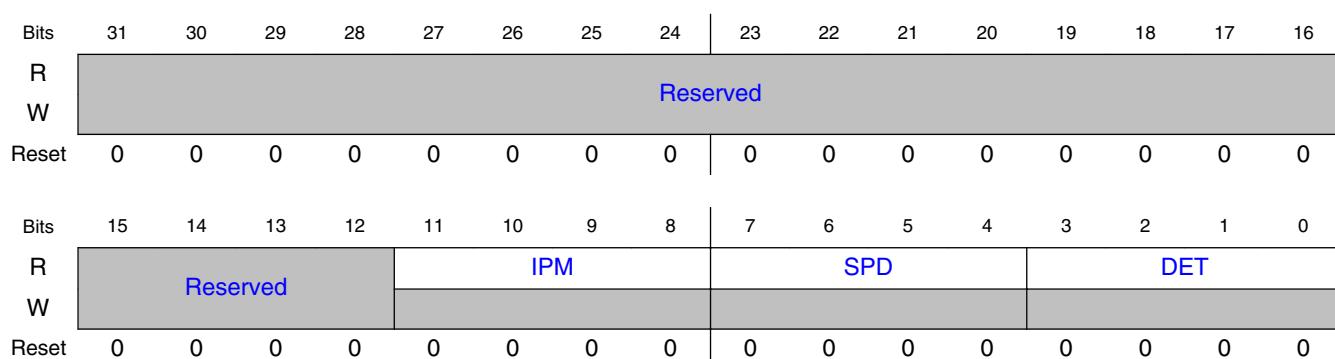
#### 32.3.29.1 Offset

Register	Offset
PxSSTS	128h

#### 32.3.29.2 Function

This register conveys the current state of the interface and host. The HBA updates it continuously and asynchronously. When the HBA transmits a COMRESET to the device, this register is updated to its reset values.

#### 32.3.29.3 Diagram



#### 32.3.29.4 Fields

Field	Function
31-12	Reserved

Table continues on the next page...

## SATA AHCI register descriptions

Field	Function
—	
11-8 IPM	<p>Interface power management</p> <p>This bit indicates the current interface state. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - Device not present or communication not established</li> <li>0001b - Interface in active state</li> <li>0010b - Interface in partial power management state</li> <li>0110b - Interface in slumber power management state</li> </ul>
7-4 SPD	<p>Current interface speed</p> <p>This bit indicates the negotiated interface communication speed. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - Device not present or communication not established</li> <li>0001b - Generation 1 communication rate negotiated</li> <li>0010b - Generation 2 communication rate negotiated</li> <li>0011b - Generation 3 communication rate negotiated</li> </ul>
3-0 DET	<p>Device detection</p> <p>This bit indicates the interface device detection and PHY state. The means by which the implementation determines device presence may be vendor specific. However, device presence must always be indicated anytime a COMINIT signal is received. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - No device detected and PHY communication not established</li> <li>0001b - Device presence detected but PHY communication not established</li> <li>0011b - Device presence detected and PHY communication established</li> <li>0100b - PHY in offline mode as a result of the interface being disabled or running in a BIST loopback mode</li> </ul>

## 32.3.30 Port x SATA control register (PxSCTL)

### 32.3.30.1 Offset

Register	Offset
PxSCTL	12Ch

### 32.3.30.2 Function

The register is used by software to control SATA capabilities. Writes to this register result in an action being taken by the host adapter or interface. Reads from the register return the last value written to it.

### 32.3.30.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R	Reserved				IPM				SPD				DET				
W																	
Reset	0	0	0	0	0	0	1	1		0	0	0	0	0	0	0	0

### 32.3.30.4 Fields

Field	Function
31-12 —	Reserved
11-8 IPM	<p>Interface power management transitions allowed</p> <p>Indicates which power states the HBA is allowed to transition to. If an interface power management state is disabled, the HBA is not allowed to initiate that state and the HBA must PMNAK<sub>P</sub> any request from the device to enter that state. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - No interface restrictions</li> <li>0001b - Transitions to the partial state is disabled</li> <li>0010b - Transitions to the slumber state is disabled</li> <li>0011b - Transitions to both partial and slumber states are disabled</li> </ul>
7-4 SPD	<p>Speed allowed</p> <p>This bit indicates the highest allowable speed of the interface. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - No speed negotiation restrictions</li> <li>0001b - Limit speed negotiation to Generation 1 communication rate</li> <li>0010b - Limit speed negotiation to a rate not greater than Generation 2 communication rate</li> <li>0011b - Limit speed negotiation to a rate not greater than Generation 3 communication rate</li> </ul>
3-0 DET	<p>Device detection initialization</p> <p>This bit controls the HBA's device detection and interface initialization. This bit may only be modified when PxCMD[ST] is 0. Changing this bit while the PxCMD[ST] bit is set to 1 results in an undefined behavior. When PxCMD[ST] is set to 1, this field should have a value of 0h.</p> <p><b>NOTE:</b> It is permissible to implement any of the SATA defined behaviors for transmission of COMRESET when DET = 1h. The settings not defined below are reserved.</p> <ul style="list-style-type: none"> <li>0000b - No device detection or initialization action requested</li> <li>0001b - Perform interface communication initialization sequence to establish communication. This is functionally equivalent to a hard reset and results in the interface being reset and communications reinitialized. While this field is 1h, COMRESET is transmitted on the interface. Software should leave the DET field set to 1h for a minimum of 1 ms to ensure that a COMRESET is sent on the interface.</li> <li>0100b - Disable the SATA interface and put PHY in offline mode</li> </ul>

### 32.3.31 Port x SATA error register (PxSERR)

#### 32.3.31.1 Offset

Register	Offset
PxSERR	130h

#### 32.3.31.2 Function

This register contains the diagnostics (DIAG) and error (ERR) bits that contains diagnostic error information and error information.

#### 32.3.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
DIAG																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

#### 32.3.31.4 Fields

Field	Function
31-16 DIAG	Diagnostic This bit contains diagnostic error information for use by diagnostic software in validating correct operation or isolating failure modes.

**Table 32-7. Diagnostic error information**

Bit	Description
31:27	Reserved

Table continues on the next page...

Field	Function	
	<b>Table 32-7. Diagnostic error information (continued)</b>	
	Bit	Description
	26	Exchanged (X): When set to 1, this bit indicates that a change in device presence has been detected since the last time this bit was cleared. The means by which the implementation determines that the device presence has changed is vendor specific. This bit must always be set to 1 anytime a COMINIT signal is received. This bit is reflected in the PxIS[PCS] bit.
	25	Unknown FIS Type (F): Indicates that one or more FISes were received by the transport layer with good CRC, but had a type field that was not recognized.
	24	Transport state transition error (T): Indicates that an error has occurred in the transition from one state to another within the transport layer since the last time this bit was cleared.
	23	Link Sequence Error (S): Indicates that one or more link state machine error conditions was encountered. The link layer state machine defines the conditions under which the link layer detects an erroneous transition.
	22	Handshake Error (H): Indicates that one or more R_ERR handshake response was received in response to frame transmission. Such errors may be the result of a CRC error detected by the recipient, a disparity or 8b/10b decoding error, or other error condition leading to a negative handshake on a transmitted frame.
	21	CRC Error (C): Indicates that one or more CRC errors occurred with the link layer.
	20	Disparity Error (D): This field is not used by AHCI.
	19	10B to 8B Decode Error (B): Indicates that one or more 10B to 8B decoding errors occurred.
	18	Comm Wake (W): Indicates that a Comm Wake signal was detected by the PHY.
	17	PHY Internal Error (I): Indicates that the PHY detected some internal error.
	16	PhyRdy Change (N): Indicates that the PhyRdy signal changed state. This bit is reflected in the PxIS[PRCS] bit.
15-0 ERR	Error	

Field	Function
	<p>This bit contains error information for use by host software in determining the appropriate response to the error condition.</p>

**Table 32-8. Error information**

Bit	Description
15:12	Reserved
11	Internal Error (E): The host bus adapter experienced an internal error that caused the operation to fail and may have put the host bus adapter into an error state. The internal error may include a master or target abort when attempting to access system memory, an elasticity buffer overflow, a primitive mis-alignment, a synchronization FIFO overflow, and other internal error conditions. Typically when an internal error occurs, a non-fatal or fatal status bit in the PxIS register is also set to give software guidance on the recovery mechanism required.
10	Protocol Error (P): A violation of the SATA protocol was detected.
9	Persistent Communication or Data Integrity Error (C): A communication error, which was not recovered, occurred that is expected to be persistent. Persistent communications errors may arise from faulty interconnect with the device, from a device that has been removed or has failed, or a number of other causes.
8	Transient Data Integrity Error (T): A data integrity error occurred that was not recovered by the interface.
7:2	Reserved
1	Recovered Communications Error (M): Communications between the device and host was temporarily lost but was re-established. This can arise from a device temporarily being removed, from a temporary loss of PHY synchronization, or from other causes and may be derived from the PhyNRdy signal between the PHY and Link layers.
0	Recovered Data Integrity Error (I): A data integrity error occurred that was recovered by the interface through a retry operation or other recovery action.

### 32.3.32 Port x command issue register (PxCI)

### 32.3.32.1 Offset

Register	Offset
PxCI	138h

### 32.3.32.2 Function

Port x command issue register

### 32.3.32.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R										CI							
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R										CI							
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### 32.3.32.4 Fields

Field	Function
31-0	Command issued
CI	<p>This bit is bit significant. Each bit corresponds to a command slot, where bit 0 corresponds to command slot 0. This bit is set by software to indicate to the HBA that a command has been built in system memory for a command slot and may be sent to the device. When the HBA receives a FIS which clears the BSY, DRQ, and ERR bits for the command, it clears the corresponding bit in this register for that command slot. Bits in this field must only be set to 1 by software when PxCMD[ST] is set to 1.</p> <p>This field is also cleared when PxCMD[ST] is written from a 1 to a 0 by software.</p>

### 32.3.33 Port x FIS-based switching control register (PxFBS)

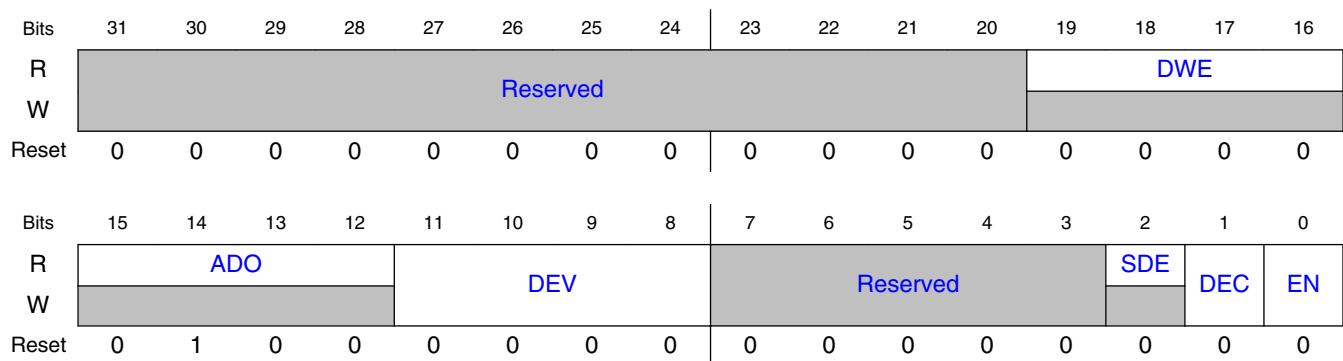
### 32.3.33.1 Offset

Register	Offset
PxFBS	140h

### 32.3.33.2 Function

This register controls and obtains the status for port multiplier FIS-based switching.

### 32.3.33.3 Diagram



### 32.3.33.4 Fields

Field	Function
31-20 —	Reserved
19-16 DWE	Device with error Set by hardware to the value of the port multiplier port number of the device that experienced a fatal error condition. This bit is only valid when PxFBS[SDE] is set to 1.
15-12 ADO	Active device optimization This bit exposes the number of active devices that the FIS-based switching implementation has been optimized for. When there are more devices active than indicated in this bit, throughput of concurrent traffic may degrade. For optimal performance, software should limit the number of active devices based on this value. The minimum value for this bit should be 2h, indicating that at least two devices may be active with high performance maintained.
11-8 DEV	Device To issue Set by software to the port multiplier port value of the next command to issue. This field enables hardware to know the port the command is to be issued to without fetching the command header.

Table continues on the next page...

Field	Function
	Software must not issue commands to multiple port multiplier ports on the same write of the PxCI register.
7-3 —	Reserved
2 SDE	<p>Single device error</p> <p>This bit is cleared on PxFBS[DEC] being set to 1 or on PxCMD[ST] being cleared to 0.</p> <p>0b - When a fatal error condition has occurred, the error applies to the entire port. To clear the error, PxCMD[ST] must be cleared to 0 by software.</p> <p>1b - When a fatal error condition has occurred, hardware believes the error is localized to one device such that software's first error recovery step should be to utilize the PxFBS[DEC] functionality.</p>
1 DEC	<p>Device error clear</p> <p>Software only sets this bit to 1 if PxFBS[EN] is set to 1 and PxFBS[SDE] is set to 1.</p> <p>0b - When hardware has completed error recovery actions, hardware clears the bit to 0. Write to 0 by software has no effect.</p> <p>1b - The HBA clears the device-specific error condition and the HBA flushes any commands outstanding for the device that experienced the error, including clearing the PxCI and PxSACT bits for that device to 0.</p>
0 EN	<p>Enable</p> <p>Software only changes the value of the EN bit when PxCMD[ST] is cleared to 0.</p> <p>0b - FIS-based switching is not being used.</p> <p>1b - A port multiplier is attached and the HBA uses FIS-based switching to communicate with it.</p>

### 32.3.34 Port 0 BIST error register (PBERR)

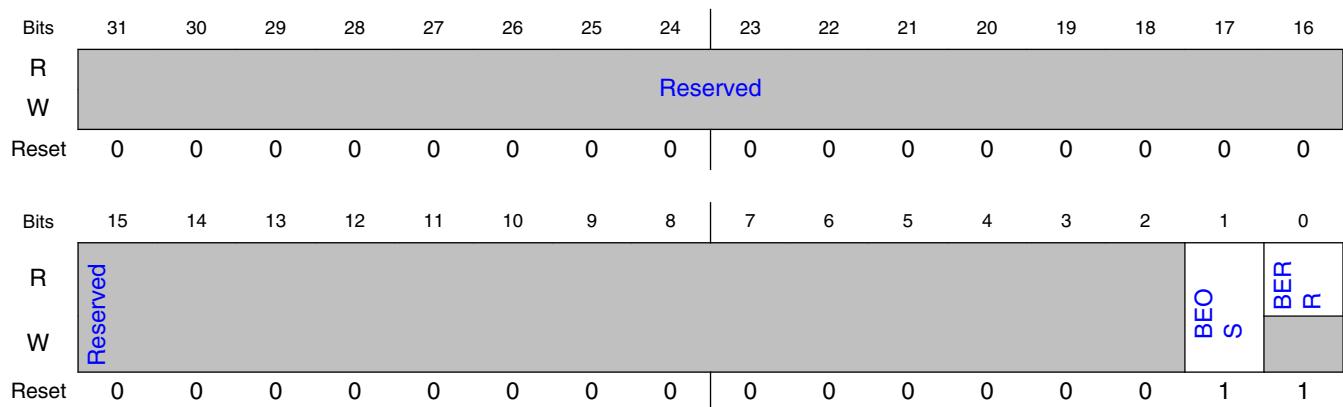
#### 32.3.34.1 Offset

Register	Offset
PBERR	170h

#### 32.3.34.2 Function

The BIST operation as programmed by SATA must report the operation as passing or failing. This register reflects the status of the BIST operation running in the link layer.

### 32.3.34.3 Diagram



### 32.3.34.4 Fields

Field	Function
31-2	Reserved
—	
1 BEOS	BIST error one shot bit 0b - BIST operation passed 1b - BIST operation failed
0 BERR	BIST error 0b - BIST operation passing 1b - BIST operation failing

## 32.4 Command layer

The operation of the command layer is defined by the AHCI specification.

### 32.4.1 Local port context management

When the AHCI controller is connected to a port multiplier supporting FIS-based switching, a local context store can be enabled to avoid the process of lookup of the related memory addressing for data transactions. This feature facilitates quick context switching and allows the AHCI to operate with multiple devices in a seamless manner. Each context stores information about the memory address and SATA block address of any executing command on the first four ports of a port multiplier.

### 32.4.2 Vendor-specific BIST operation

As part of the host self-diagnostic operation, a vendor-specific BIST mode is supported. This mode, in conjunction with a SerDes that supports serial loop-back, allows for the test of the host controller operation. When programmed, the host fetches a command from memory loop that commands FIS through the transport and the link layers and then posts the payload to the receive FIS area for checking. This mode exercises the following paths.

- DMA controller FIS transmission
- Command layer FIS transmission
- Transport layer TX FIFO FIS transmission
- Link layer FIS transmission
- PHY modes
- Link layer FIS reception
- Transport layer Rx FIFO and FIS reception
- Command layer FIS reception
- Host DMA controller FIS reception

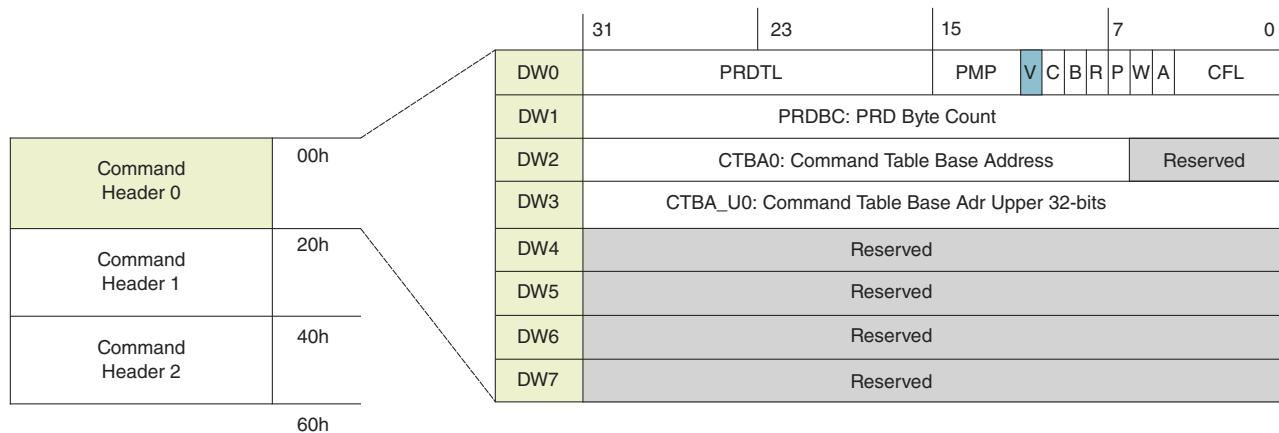
To run this self-test, the software needs to perform the following operation:

1. Build the vendor-specific header and descriptor as detailed out in the Vendor Specific BIST Header and Descriptor. For more information, refer [Table 32-10](#).
2. Place the PHY in the loop-back mode. For more information, refer to SerDes module chapter for the PHY loop.
3. Set PPCFG[FPR] to 1. Issue this command to the host controller.

When the host controller indicates the command has completed, the software examines the contexts of the command descriptors stats field. If the preprogrammed values are present, it considers the test passed. Vendor-specific BIST header, descriptor, and register settings are described in the subsequent sections.

#### 32.4.2.1 Command list structure

To support the vendor BIST operation, the command header structure has been modified. The DW0[11] bit has been designated as VBIST. Setting this bit indicates that the associated command is to be used as the payload for a vendor BIST operation.

**Figure 32-2. Command list structure**

This table describes the fields in the command header.

**Table 32-9. DW 0 - Description information**

Bit	Description
31:12	Per the AHCI specification
11	VBIST Setting this bit indicates that the associated command is to be used as payload for a vendor BIST operation.
10:0	Per the AHCI specification

**Table 32-10. Vendor BIST command header**

DWord number	Hexadecimal value
DW0	32'h0000_0805
DW1	32'h0000_0000
DW2	System Address (usage dependent)
DW3	System Address (usage dependent)

### 32.4.2.2 Vendor BIST command descriptor command FIS

The A and B values can be filled in by the user according to the pattern that needs to be tested.

**Table 32-11. Command descriptor command**

DWord number	Hexadecimal value
DW0	32'h0001_0058
DW1	32'hAAAA_A034
DW2	32'hBBBB_B034
DW3	Reserved - 32'h0000_0000
DW4	Reserved - 32'h0000_0000
DW5	Reserved - 32'h0000_0000
DW6	Reserved - 32'h0000_0000
DW7	Reserved - 32'h0000_0000

### 32.4.2.3 Receive FIS area reg D2H RFIS structure

The table below provides the receive FIS area reg D2H RFIS structure.

**Table 32-12. Receive FIS area reg D2H RFIS structure**

DWord number	Hexadecimal Value	Hexadecimal Value
DW0	32'hAAAA_A034	32'hBBBB_B034
DW1	32'hBBBB_B034	32'hAAAA_A034
DW2	32'hAAAA_A034	32'hBBBB_B034
DW3	32'hBBBB_B034	32'hAAAA_A034
DW4	32'hAAAA_A034	32'hBBBB_B034

## 32.5 PhyControl BIST modes

The BIST modes within the PhyControl layer are designed to cover two main functions.

- Verification of the interface between PhyControl (controller layer) and SerDes
- Generation of limited patterns for compliance testing

The following programming steps are performed when the system software places the PhyControl into one of its BIST modes:

1. Initialize the Phy1Cfg register to place the system into a state of Force Ready and Near End Retimed Loopback, to enable PhyControl BIST Pattern and PhyControl BIST Clear Error, and to select the required BIST pattern .
2. The software brings the device online. This operation causes the controller to de-assert the reset to the PHY and allow the setup start.
3. The software inserts a delay of 10 seconds before proceeding. This is to allow the setup to stabilise and the Rx part of the BIST checker lock to the incoming pattern.
4. Once the delay has elapsed, the software clears the PhyControl BIST Clear Error bit of the register.
5. When these steps are completed, the software monitors the BIST error bit of the Port Vendor Specific registers 0. If this bit remains low the PhyControl passes BIST. If this bit is set then the PhyControl fails BIST. The bit remains set until the software clears it using the PhyControl BIST Clear Error bit of the Phy1Cfg register.

The PhyControl layer supports a number of test patterns. Each pattern is applied in an un-encoded format to the PHY which encodes 8b/10b , serialises, loopbacks, de-serialises and decodes 10b/8b.

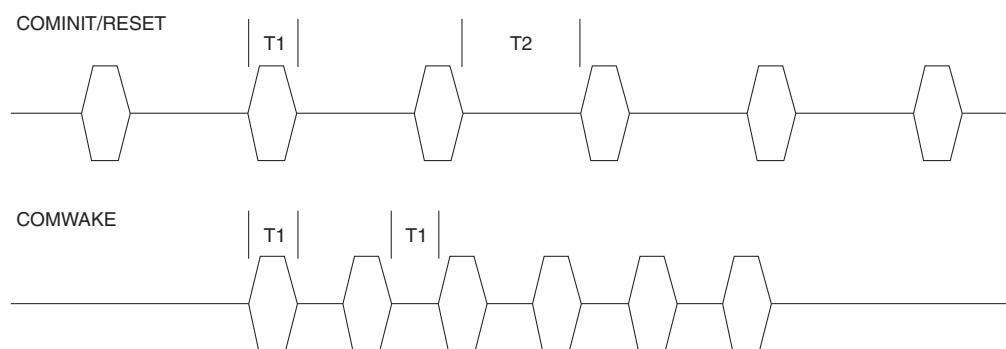
- LBP (Lone bit pattern): The lone-bit pattern is designed to be used to generate a pattern only.
- LFTP (Low frequency test pattern): This pattern provides a low frequency, which is allowed within the Serial ATA encoding rules. Pattern 32'hD30.3\_D30.3\_D30.3\_D30.3. The pattern is repetitive.
- MFTP (Mid frequency test pattern): This pattern provides a middle frequency which is allowed within the Serial ATA encoding rules. Pattern encoded 32'hD24.3\_D24.3\_D24.3\_D24.3. The pattern is repetitive.
- HFTP (High frequency test pattern): This pattern provides the maximum frequency allowed within the Serial ATA encoding rules. Pattern encoded 32'hD10.2\_D10.2\_D10.2\_D10.2. The pattern is repetitive.
- PRBS Pattern: This pattern is a long running 32 bit PRBS pattern initially seeded to 32'h1234\_5678; The PRBS algorithm is defined as: {PRBSTest[30:0], (PRBSTest[31] ^ PRBSTest[30] ^ PRBSTest[29] ^ PRBSTest[28] ^ PRBSTest[27] ^ PRBSTest[21])};
- BIST pattern: This pattern provides the counting pattern that allows maximum combinations of characters be tested. It consists of Pattern:
  - 32'hD0.0\_D1.0\_D2.0\_D3.0
  - 32'hD4.0\_D5.0\_D6.0\_D7.0
  - 32'hD8.0\_D9.0\_D10.0\_D11.0
  - 32'hD12.0\_D13.0\_D14.0\_D15.0
  - 32'hD28.7\_D29.7\_D30.7\_D31.7

BIST pattern transmission.

1. Seven ALIGN primitives are transmitted to force the Rx path to lock to the transmitted data.
2. A special lock primitive 32'h4A\_4A\_4A\_7C is transmitted to start the pattern match on the Rx side.
3. The selected pattern is transmitted for 256 byte intervals and then the circuit starts from state 1.

## 32.6 PHY control configuration register OOB timing setup

In SATA systems three out of band (OOB) signals COMRESET, COMINIT, and COMWAKE are used during link initialization. These are achieved by transmission of either a burst of four Gen1 ALIGNp primitives or a burst composed of sixteen D24.3 characters, each burst having a duration of 160 UIOOB (106.67 ns). The individual bursts are followed by idle periods (at common-mode levels), having durations as depicted in the figure and the table below. The duration of the idle period is used to decode the type of OOB signal.



**Figure 32-3. OOB signaling timing**

Parameter	Units	Limit	Specification
UIOOB, UI During OOB Signaling	Ps	Nominal	666.67
COMINIT/COMRESET and COMWAKE Transmit Burst Length	Ns	Nominal (T1)	106.7
COMINIT/COMRESET Transmit Gap Length	Ns	Nominal (T2)	320.0
COMWAKE Transmit Gap Length	Ns	Nominal (T1)	106.7

*Table continues on the next page...*

## Modifications to the AHCI standard PRDT entry

COMWAKE Gap Detection Windows	Ns	May Detect Shall detect Shall not detect	$35 \leq T < 175$ $101.3 \leq T \leq 112$ $T < 35 \text{ or } T \geq 175$
COMINIT/COMRESET Gap Detection	Ns	May Detect Shall detect Shall not detect	$175 \leq T < 525$ $304 \leq T \leq 336$ $T < 175 \text{ or } T \geq 525$

## 32.7 Modifications to the AHCI standard PRDT entry

In order to support PRDT control of the AW/RCACHE, the DW2 field of the standard PRDT entry has a new bit designation as follows:

**Table 32-13. DW 2 – PRDT Re-designation for AXI operation**

Bit	Description
31:00	Reserved
Re-designation	
31:18	Reserved
17	Reserved
16	CACHE bits controlled from PRD (Sub Control)
15:11	Reserved
10:8	Reserved
7:4	This is the value driven onto A[R/W]CACHE when the AXI master is posting a data burst read / write address associated with this PRD entry to the memory controller
3:2	Reserved
1	Inject a CRC error into the associated Received Data FIS
0	Inject a CRC error into the associated Transmitted Data FIS

When a PRDT entry is processed to move data through the AXI master, first the global control field of the AXI Cache Control register bits 12 and 28 are examined to determine the values for the CACHE control signals. If the global control indicates that the values should be controlled from the PRDT then the sub control determines which values control the CACHE signals.

**Table 32-14. Global control field**

Global	Sub	AR/WCACHE
0	0	AXI Cache Control Register
0	1	AXI Cache Control Register

*Table continues on the next page...*

**Table 32-14. Global control field (continued)**

Global	Sub	AR/WCACHE
1	0	AXI Cache Control Register
1	1	DW 2 – PRDT Re-designation for AXI operation

## 32.8 Transport layer architectural overview

The function of the SATA transport layer is to interface between the command and link layers in the transmission and reception of the Frame Information Structures (FIS).

During transmission, the transport layer frames the FIS placed into the TX FIFO. The FISs are framed based on a programmed length for non-data FIS and/or a configurable length for data FIS. When the transport layer is instructed to send a non-data FIS, it employs a retry policy until the far end signals accepts the transmitted FIS.

On reception, the transport layer de-frames the FISes and places these into the RX FIFO. When an FIS is received, the transport layer informs the command layer.

For a non-data FIS, the FIS is considered received when an EOF is signaled by the link layer and FIS is received with a good CRC.

For a short vendor-specific FIS, the FIS is considered as a non-data FIS.

For longer vendor-specific FIS, the FIS reception is signaled when the RX FIFO reaches its water mark.

For a data FIS, the FIS is considered received when the First DWord (header) is written into the FIFO.

The transport layer is responsible for crossing the clock domain between the transport layer txDWord and the rxDWord clocks and command layer clock domain. The receive FIFO is written to the transport layer receive DWord clock with data contained in the FIS sent by the link layer. Once the data is stable at the output of the receive FIFO on the command layer clock domain, the command layer can take the data. If the command layer is not ready to accept the data, the data builds up in the receive FIFO. When the receive FIFO exceeds its threshold, the transport layer stalls the link layer, which in turn sends HOLD primitives to the far end to stall it. This threshold takes into consideration the latency involved in getting the far end to stop transmitting the data. This threshold is programmable to allow for the use of high latency repeaters or re-timers in between the host and the device.

The transmit FIFO is written to the command layer clock, with data to be sent in the FIS transferred by the DMA controller. Once the data is stable at the output of the transmit FIFO on the transmit DWord clock domain, the link layer can take the data. If the transmit FIFO cannot supply data to the link layer, the transport layer stalls the link layer, which in turn sends HOLD primitives to the far end to stall it.

## **32.9 Link layer overview**

### **32.9.1 General operation**

The function of the SATA link layer is to interface between the transport and PhyCtrl layers in the transmission and reception of frames and primitives. The link layer utilizes the two unidirectional links provided by the SATA interface to maintain coordinated communication between the host and the device. Payload data can only be transmitted in one direction at a time.

On transmit, the link layer first communicates with the peer far end link layer to determine if it is ready to receive. Assuming the far end link layer can receive data, the local link layer begins to take data in the form of DWords from its transport layer. It inserts Start-of-Frame (SOF) before the start of the data portion of a frame, calculates and inserts the CRC after the data portion of a frame, and inserts the End-of-Frame (EOF) primitive at the end. The link layer scrambles the contents of the frame, including the calculated CRC, but excluding the SOF and EOF delimiters, and any other embedded primitives. The 8B/10B encoding of the data is done either in the PHY control layer or in the PHY layer, depending on the nature of the PHY employed. At the end of the transmission, the link layer reports transmission status to the transport layer.

On receive, the link layer first acknowledges its readiness to receive its peer link layer. Then it awaits reception of the SOF primitive that marks the start of the received data. Following detection of the SOF primitive, the link layer proceeds to accept the incoming data. The 8B/10B decoding of the data is done either in the PHY control layer or in the PHY layer. After this, the link layer removes all primitives including the SOF and EOF delimiters. It then descrambles the contents of the frame. The link layer also calculates the CRC on the incoming frame between the SOF and EOF delimiters, and compares this calculated value to the received value. Any mismatch is reported to the transport layer. During frame reception, disparity or code errors are reported to the command layer, and appropriate action is taken in the link layer. The descrambled and decoded receive data stream is passed to the transport layer as the frame is being received. Finally at the end of the frame, the link layer reports reception status to the transport layer.

The link layer also partakes in flow control between the local and remote ends. The layer supports flow control actions based on the local FIFO status (which is located in the transport layer), or in response to receiving flow control messages from the remote end.

The transmit side of the link layer is also responsible for inserting a pair of ALIGN primitives every 254 DWords, or more frequently, if programmed by the user.

### **32.9.2 List of functions**

The link layer is composed of a number of functions, as listed below:

- Link layer state machines
- Frame content scrambler and descrambler
- CRC generation and checking
- Bus interfaces to PhyCtrl and transport layer
- CONT primitive processing
- ALIGN insertion on transmit
- Debug functionality
- BIST support

### **32.9.3 Link layer state machines**

Functionally, there are four link layer state machines. These are listed below and described in the subsequent sub-sections:

- Link idle state machine
- Transmit state machine
- Receive state machine
- Power mode change state machine

#### **32.9.3.1 Link idle state machine**

The link idle state machine performs the following functions:

- It is responsible for detecting a transmit request from the transport layer, or a frame reception request from the far end, and arbitrating if these two events coincide. The SATA specification defines that the host end always backs down in this case.

- It interprets power mode change requests from both transport and PhyCtrl layers, and initiates actions to enable the power mode change.
- It also detects negation and assertion of PHY\_READY from the PhyCtrl layer and notifies the transport layer of the change.

### 32.9.3.2 Transmit state machine

The transmit state machine manages frame transmission to the PhyCtrl layer. The state machine places the Start-of-Frame (SOF) and End-of-Frame (EOF) headers on each frame, calculates the CRC and inserts it before the EOF delimiter. Between the SOF and CRC markers, the link layer accepts the current DWord from the transport layer and uses this as the next DWord of the frame. The link layer also inserts a pair of ALIGN primitives every 254 DWords of the frame data. Finally, at the end of the frame transmission, the state machine waits for status from the far end link layer through the received R\_OK or R\_ERR primitives. If the far end receives the frame correctly, the local link layer signals TX\_OK to the transport layer; else, it signals TX\_NOT\_OK to the transport layer.

This state machine also partakes in the flow control actions if necessary, during packet transmission. If the transport layer cannot supply a new DWord and the frame is not finished, the transmit state machine responds by sending HOLD primitives until such time as the transport layer is ready with valid frame data. Also, during frame transmission, if the state machine detects a received HOLD primitive from the PHY Layer, it interrupts the current frame transmission and sends HOLDA primitives to the PHY to be transmitted to the far end.

The current frame transmission can only be aborted by two events.

- On reception of a DMAT primitive from the far end. In this case, the link layer state machine stops the current transfer and calculates and inserts the current CRC. This is a controlled termination.
- When the transport layer wishes to send a control register frame, signaled through TRANSMIT\_CRF.

If at any point in the frame transmission process, the link layer detects error conditions, it signals these to the command layer. The errors can occur if:

- PHY\_READY negates
- The link layer receives SYNC primitives during frame transmission

### 32.9.3.3 Receive state machine

The receive state machine is responsible for frame reception from the PHY layer. It removes the Start-of-Frame (SOF) and End-of-Frame (EOF) headers and other primitives from each frame, calculates the CRC and compares it to the received CRC. Between the SOF and CRC markers, the link layer accepts the current DWord from the PhyCtrl layer and uses this as the next DWord of the frame, transferring it to the transport layer. At the end of the frame reception, if the calculated CRC is not the same as the received CRC, the link layer signals an error to the transport layer. This is done through RX\_CRC\_OK and RX\_CRC\_NOT\_OK statuses. During frame reception, if no errors are detected, the link layer transmits R\_IP primitives to the far end peer link layer. Finally, at the end of the frame reception, the link layer sends the R\_OK primitive if no error is detected during reception. If an error is detected, it sends an R\_ERR primitive instead.

The receive state machine also partakes in flow control actions if necessary, during FIS reception. If the transport layer cannot accept a new DWord, (because its receive FIFO has reached its watermark level), and the FIS is not finished, the receive state machine responds by sending HOLD primitives on the back channel until such time as the transport layer is ready to accept FIS data again. Also, during FIS reception, if the state machine detects a received HOLD primitive from the far end, it responds by sending HOLDA primitives to the far end.

The current frame reception can be interrupted if the transport layer wishes to send a control register frame, signaled through TRANSMIT\_CRF.

If at any point in the frame reception process, the link layer detects error conditions, it signals these to the command layer. The errors can occur if

- PHY\_READY negates
- The link layer receives SYNC primitives
- The link layer receives WTRM primitives before EOF

### 32.9.3.4 Power mode change state machine

This state machine handles change of power mode requests. These requests can come from the command layer Super Set registers or the far end. The command layer signals a change request by asserting CFG\_LINK\_GO\_PARTIAL or CFG\_LINK\_GO\_SLUMBER. This state machine responds by transmitting PMREQ\_P/PMREQ\_S primitives to the far end and waiting for PMACK primitives from it in response. Once PMACK is received, the state machine instructs the PHY layer to enter either partial or slumber state, by driving PHY\_GO\_PARTIAL or PHY\_GO\_SLUMBER signals. These signals must remain active for the duration of the new power mode.

A write of '1' to the CFG\_LINK\_WAKEUP bit or reception of a COMWAKE from the far end will initiate a resume to active power mode.

**Note:** CFG\_LINK\_WAKEUP should be set back to '0' once PhyReady is asserted again.

If the link layer receives a PMREQ\_P/PMREQ\_S primitive from a peer link layer, and is enabled to perform power management modes (CFG\_EN\_PARTIAL/SLUMBER = '1'), it responds by sending at least four PMACK primitives, and then asserts PHY\_GO\_PARTIAL or PHY\_GO\_SLUMBER to the PhyCtrl block. A write of '1' to the CFG\_LINK\_WAKEUP bit or reception of a COMWAKE from the far end initiates a resume to the active power mode.

**Note:** CFG\_LINK\_WAKEUP should be set back to '0' once PhyReady is asserted again.

If the link layer receives an XRDY primitive from the far-end while it is in the partial or slumber state, it returns to idle and signals a link sequence error to the command layer, that is LINK\_SEQ\_ERROR = 1. Automatic partial to slumber transitions can be set by programming the CAP2 register (offset 24h)

### **32.9.4 Frame content scrambler and descrambler**

Two separate scramblers are used in serial ATA - one for the data payload and another for repeated primitive suppression.

The contents of each DWord of data (excluding all primitives) between SOF and EOF must be scrambled before 8B/10B encoding. Scrambling is performed on DWord quantities according to the following polynomial:

$$G(X) = X^{16} + X^{15} + X^{13} + X^4 + 1$$

The scrambler is initialized with a seed value of 16'hFFFF at each SOF transmission and rolls over every 2048 DWords. Payload data is scrambled prior to transmission, by XORing the data to be transmitted with the output of this scrambler.

If a CONT primitive is transmitted, the intervening data between the last CONT primitive and a subsequent primitive must be scrambled as well. This scrambler uses the same polynomial as defined above for data payload scrambling and is reset to the initial value upon detection of a COMINIT or COMRESET event (.PHY\_GOT\_CIOR = '1'). If a CONT primitive is transmitted or received during a frame transfer, the current data payload scrambler value at the last DWord is held.

When payload data is received by the link layer it is de-scrambled by XORing it with the output of its descrambler. The descrambler is re-seeded at the beginning of the received data payload, that is at each SOF reception. The descrambler uses the same polynomial as the scrambler.

### 32.9.5 CRC generator and checker

A 32 bit CRC is calculated on the data contents of each frame, and is inserted in the DWord before the EOF. The CRC covers all data bytes in the frame excluding any primitives, such as SOF, EOF, HOLD, HOLDA, DMAT, SYNC, X\_RDY, R\_RDY or ALIGNs.

The CRC generator works on DWord quantities. Any padding to the boundary is done in the transport layer. The polynomial used for the CRC is as follows:

$$G(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

The CRC is initialized with a seed value of 32'h52325032 at each SOF.

The CRC generation or checking does not apply to primitives (as stated above) or to CONT'ed primitives. If a CONT primitive is transmitted or received, the intervening data between the last CONT primitive and a subsequent primitive is not included in the CRC calculation for a frame. If this happens during a frame transfer, the current CRC scrambler value at the last DWord is held.

### 32.9.6 8B/10B encode and decode

All data and primitives must be encoded prior to transmission on the line. The standard implementation has the 8B/10B encoder and decoder located in the PHY control layer.

### 32.9.7 CONT primitive processing

The link layer is capable of replacing repetitive primitive streams with scrambled data, by use of the CONT primitive. This reduces EMI emissions, as primitives are not scrambled.

The link layer can transmit a CONT primitive at a point where it knows it must transmit a number of repeating primitives. After a CONT primitive has been transmitted, the link layer transmits scrambled junk data to the PHY layer. The content of this junk data is disregarded. At the far end link layer, the reception of a CONT primitive implies that the last received valid primitive will be repeated until it receives the next valid non-align primitive. Transmission of a new valid primitive halts the current CONT processing; reception of a new valid non-align primitive halts the current CONT processing.

This action can occur on transmit and receive. The link layer supports both transmission and reception of CONT primitives. The scrambler and the CRC generator/checker actions are changed during the transmission and reception of CONT sequences as described in [Frame content scrambler and descrambler](#) and [CRC generator and checker](#).

Transmission of CONT primitives are not allowed until the link layer state machine has initialized correctly, and has transmitted 16 non-align primitives.

For diagnostic purposes CONT insertion can be suppressed in the transmit path; the receive path can always receive those.

### **32.9.8 ALIGN insertion**

The link layer is responsible for ALIGN insertion and removal at a fixed frequency. A pair of ALIGN primitives are inserted into the transmit data stream every 254 DWords. These primitives are used in the PHY layer to maintain character alignment and to help in elastic buffering, if required.

At the receive end, the ALIGN primitives are stripped from the incoming data stream in the link layer.

For diagnostic purposes, the rate of ALIGNS can be increased as much as two ALIGNS per one DWord, that is: ALIGN, ALIGN, data, ALIGN, ALIGN.

Also, the CFG\_SEND\_4\_ALIGNS bit can be set to instruct the link layer to send four aligns at a time, instead of two.

### **32.9.9 Debug functionality**

There are a number of useful features designed into the link layer to aid debug. These include:

- A configuration bit in the command layer (CFG\_TX\_BADCRC) can be set to force the link layer to transmit a bad CRC at the end of a frame. One bad CRC is transmitted every time the bit is configured. To force transmission of a new bad CRC, the configuration bit needs to be cleared and written to again.
- A configuration bit in the command layer (CFG\_RX\_BADCRC) can be set to force the link layer to detect a bad CRC value at the end of a frame. Only one bad CRC is detected for one setting. The configuration bit must be cleared and written to again to force detection of another bad CRC.

- A configuration bit in the command layer (CFG\_TXPRIM\_JUNK) can be set to force the junk data that is transmitted after a CONT primitive to be equal to 32'hDEADBEEF.
- The align insertion rate can be increased using the CFG\_ALIGN\_RATE register field in the command layer.
- Four error counters can be monitored by issuing register reads to the command layer. These error counters include: Disparity error counter, Code error counter, PHY Internal error counter and Control Character error counter (see signal descriptions for detail).
- A number of configuration bits in the command layer can be used to override normal primitive insertion. For example,
  - Set CFG\_PRIM\_OVR\_STATE = the L\_SendHold state (16)
  - Set CFG\_PRIM = 32'hb5b5957c , that is a SYNC primitive
  - During the transfer, set CFG\_PRIM\_OVR\_EN = 1. When the link layer detects a rising edge on CFG\_PRIM\_OVR\_EN, it inserts one SYNC primitive into the datastream in place of the HOLD, when the LINK\_STATE reaches the L\_SendHold state. Only one HOLD primitive is overridden - the CFG\_PRIM\_OVR\_EN must be cleared and written to again to force another override to occur.

### 32.9.10 BIST support

The transmit and receive sub-blocks of the link layer contain logic to support BIST activate FIS functionality.

When a BIST activate FIS is either received or transmitted successfully by the transport layer, it issues a request to the link layer to enter BIST mode ( LINK\_BIST\_REQ = 1). This forces the link layer to enter a BIST state in its state machine as soon as it receives a SYNC primitive from the far-end. When the BIST state is entered, an acknowledge signal is sent to the transport layer, that is LINK\_BIST\_ACK = 1. In the BIST state, the link layer transmits a data sequence as specified by the two BIST data patterns in the BIST activate FIS. The link layer also monitors the incoming data from the PhyCtrl block to detect if the BIST data pattern is the same as specified in the BIST activate FIS. When it detects the correct data sequence, the BIST\_ERR output is de-asserted. This signal stays de-asserted unless an error occurs in the datastream from the far-end. The BIST\_ERR output is sent to the command layer from where it can be read by the host software. The link layer also outputs a signal called BIST\_ACTIVE to the command layer. This singnal is asserted when the link layer enters its BIST state and de-asserted when BIST mode is exited due to a COMRESET being received.

## 32.10 PHY control layer overview

The PHY control layer operates between the SerDes and link layers.

The main functions of the PHY control layer are

- Data path operation
  - Rx data path
  - Tx data path
- PHY Initialization state machine
  - Out of band processing
  - Speed negotiation

On receive, the PHY Control layer converts the encoded 20-bit or decoded 16-bit parallel data from the SerDes to a 32-bit DWord, which it presents to the link layer. The PHY control layer aligns the control word of the SATA primitive to the lowest word position of the DWord. The two main configurations of the PHY control layer operate on either encoded data or decoded data, as sourced from the SerDes. In the case of encoded data, the PHY control contains an 8b/10b decoder function and decodes the incoming data into data, control/data and code or disparity error. In the case of decoded data, the PHY control sources data, control/data and code or disparity error from the SerDes.

The receive DWord clock is generated from the receive core clock outputted by the SerDes. This is generated as part of the DWord alignment process.

On transmit, the PHY control layer takes in the 32-bit transmit data from the link layer and converts the data into encoded 20-bit or un-encoded 16-bit parallel data from the SerDes. The control/data bit from the link layer (which is always assumed to be associated with the lowest byte position of the transmit DWord) is also passed onto the SerDes with the appropriate word. The PHY control layer takes the transmit word clock outputted by the SerDes and converts it to a DWord transmit clock which it sends to the link layer.

The PHY control layer implements the PHY initialization state machine. The state machine implements the device PHY initialization state machine as defined in the Serial ATA Revision 3.1 June 2, 2009.

## 32.11 Reset

The SATA AHCI controller supports three levels of reset:

- Software reset - a single device on one of the ports is reset, but the HBA and physical communication remain intact. This is the least intrusive.
- Port reset – the physical communication between the HBA and device on a port are disabled. This is more intrusive.
- HBA reset – the entire HBA is reset, and all ports are disabled. This is the most intrusive.

When an HBA or port reset occurs, PHY communication is re-established with the device through a COMRESET followed by the normal out-of-band communication sequence defined in SATA. At the end of the reset, the device, if working properly, will send a D2H Register FIS, which contains the device signature. When the HBA receives this FIS, it updates the PxTFD[STS] and PxTFD[ERR] bits, and updates the PxSIG register with the signature.

Software is recommended to follow a staggered reset approach, starting from a less intrusive reset mechanism and then using a more intrusive reset mechanism only if the less intrusive mechanism does not succeed in clearing the condition.

### 32.11.1 Software reset

Legacy software contained a standard mechanism for generating a reset to a SATA device – setting the SRST (software reset) bit in the Device Control register. SATA has a more robust mechanism called COMRESET, also referred to as port reset. A port reset is the preferred mechanism for error recovery and should be used in place of software reset.

To issue a software reset in AHCI, software builds two H2D Register FISes in the command list. The first Register FIS has the SRST bit set to '1' in the Control field of the Register FIS, the 'C' bit is set to '0' in the Register FIS, and the command table has the CH[R] (reset) and CH[C] (clear BSY on R\_OK) bits set to '1'. The CH[R] (reset) bit causes the HBA to perform a SYNC escape if necessary to put the device into an idle condition before sending the software reset. The CH[C] (clear BSY on R\_OK) bit needs to be set for the first Register FIS to clear the BSY bit and proceed to issue the next Register FIS since the device does not send a response to the first Register FIS in a software reset sequence. The second Register FIS has the SRST bit set to '0' in the Control field of the Register FIS, the 'C' bit is set to '0' in the Register FIS, and the command table has the CH[R] (reset) and CH[C] (clear BSY on R\_OK) bits cleared to '0'.

Refer to the Serial ATA Revision 2.6 specification for more information on the software reset FIS sequence.

**Table 32-15. Register FIS host to device SRST On**

**Table 32-16. Register FIS host to device SRST Off**

When issuing a software reset sequence, there should not be other commands in the command list. Before issuing the software reset, software must clear PxCMD[ST], wait for the port to be idle (PxCMD[CR] = '0'), and then re-set PxCMD[ST].

## Reset

PxTFD[STS[BSY]] and PxTFD[STS[DRQ]] must be cleared prior to issuing the reset. If PxTFD[STS[BSY]] or PxTFD[STS[DRQ]] is still set based on the failed command, then a port reset should be attempted or command list override (PxCMD[CLO]) should be used if supported. Note that a device-to-host FIS from a previously failed command may be received after the PxCMD[ST] bit has been cleared and/or re-set. It is recommended that the HBA ignore such a FIS or SYNC Escape that FIS (as directed by the CH[R] bit being set in the first software reset H2D Register FIS).

### 32.11.2 Port reset

If a port is not functioning properly after a software reset, software may attempt to re-initialize communication with the port through a COMRESET. It must first clear PxCMD[ST] and wait for PxCMD[CR] to clear to '0' before re-initializing communication. However, if PxCMD[CR] does not clear within a reasonable time (500 milliseconds), it may assume the interface is in a hung condition and may continue with issuing the port reset.

Software causes a port reset (COMRESET) by writing 4'h1 to the PxSCTL[DET] bit to invoke a COMRESET on the interface and start a re-establishment of PHY layer communications. Software shall wait at least 1 millisecond before clearing PxSCTL[DET] to 4'h0; this ensures that at least one COMRESET signal is sent over the interface. After clearing PxSCTL[DET] to 4'h0, software should wait for Serial ATA AHCI 1.3 Specification 101 communication to be re-established as indicated by bit 0 of PxSSTS[DET] being set to 4'h1. Then software should write all 1s to the PxSERR register to clear any bits that were set as part of the port reset.

When PxSCTL[DET] is set to 4'h1, the HBA shall reset PxTFD[STS] to 8'h7F and shall reset PxSSTS[DET] to 4'h0. When PxSCTL[DET] is set to 4'h0, upon receiving a COMINIT from the attached device, PxTFD[STS[BSY]] shall be set to 1'h1 by the HBA.

### 32.11.3 HBA reset

If the HBA becomes unusable for multiple ports, and a software reset or port reset does not correct the problem, software may reset the entire HBA by setting GHC[HR] to '1'. When software sets the GHC[HR] bit to '1', the HBA shall perform an internal reset action. The bit shall be cleared to '0' by the HBA when the reset is complete. A software write of '0' to GHC[HR] shall have no effect. To perform the HBA reset, software sets GHC[HR] to '1' and may poll until this bit is read to be '0', at which point software knows that the HBA reset has completed.

If the HBA has not cleared GHC[HR] to '0' within 1 second of software setting GHC[HR] to '1', the HBA is in a hung or locked state.

When GHC[HR] is set to '1', GHC[AE], GHC[IE], the IS register, and all port register fields (except PxFB/PxFBU/PxCLB/PxCLBU) that are not HwInit in the HBA's register memory space are reset. The HBA's configuration space and all other global registers/bits are not affected by setting GHC[HR] to '1'. Any HwInit bits in the port specific registers are not affected by setting GHC[HR] to '1'. The port specific registers PxFB, PxFBU, PxCLB, and PxCLBU are not affected by setting GHC[HR] to '1'. If the HBA supports staggered spin-up, the PxCMD[SUD] bit will be reset to '0'; software is responsible for setting the PxCMD[SUD] and PxSCTL[DET] fields appropriately such that communication can be established on the SATA link. If the HBA does not support staggered spin-up, the HBA reset shall cause a COMRESET to be sent on the port.

**Reset**

# Chapter 33

## SerDes Module

### 33.1 The SerDes module as implemented on the chip

This section provides more details about how the SerDes module is implemented on this chip.

#### NOTE

- The LS1043A implements a PCI Express Gen 2 controller with link speeds up to 5 Gbps.
- All the Ethernet interfaces of LS1043A support full-duplex mode only; any references to half-duplex mode should be ignored.

#### 33.1.1 SerDes lane assignments and multiplexing

##### 33.1.1.1 Configuration of networking SerDes

The following table shows the available networking protocols for the SerDes module. Correct configuration involves selecting the correct values for the SerDes RCW fields (see the RCW fields in the Reset, Clocking, and Initialization chapter).

- SerDes 1 Configuration:
  - Protocol(s): Selected using RCW[SRDS\_PRTCL\_S1]
  - PLLs: Enabled using RCW[SRDS\_PLL\_PD\_S1]
  - PLL Reference Clock: RCW[SRDS\_PLL\_REF\_CLK\_SEL\_S1]

### 33.1.1.2 SerDes protocols

The following table shows the supported protocol options for the SerDes module. The following notation conventions are used in the table:

- SGMII notation for frame manager (FMan):
  - sg.mn means SGMII (1 lane @ 1.25 Gbps or 3.125 Gbps)
  - "m" indicates that MAC is from FMan.
  - "n" indicates which MAC on the FMan.
  - For example, "sg.m9," indicates SGMII for MAC 9 on FMan.
- QSGMII notation for frame manager (FMan)
- XFI notation for FMan:
  - xfi.mn means XFI (1 lane @ 10.3125 Gbps)
  - "m" indicates that MAC is from FMan.
  - "n" indicates which MAC on the FMan.
  - For example, "xfi.m9," indicates XFI for MAC 9 on FMan.
- PCI Express :
  - PEXn (5/2.5) means PCI express operating up to 5 or 2.5 Gbps depending on maximum rate selection and training.
- SATA
  - SATAn (6/3/1.5) means SATA operating at 6 or 3 or 1.5 Gbps depending on rate selection. The rate of selection is performed by PxSCTL[SPD] register as described in SATA 3.0.

**Table 33-1. Supported SerDes options**

SRDS_PRTCL_S1 RCW[128:143] (hex)	A	B	C	D	PLL Mapping
0000	Unused				
1555	xfi.m9	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x1)	1222
2555	sg.m9 (2.5G)	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x1)	1222
4555	qs.m1,2,5,6	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x1)	1111
4558	qs.m1,2,5,6	PCIe 1 (x1)	PCIe 2 (x1)	SATA	2221
1355	xfi.m9	sg.m2	PCIe 2 (x1)	PCIe 3 (x1)	1222
2355	sg.m9 (2.5G)	sg.m2	PCIe 2 (x1)	PCIe 3 (x1)	1222
3335	sg.m9	sg.m2	sg.m5	PCIe 3 (x1)	1111
3360	sg.m9	sg.m2	PCIe 3 (x2)		1111
3355	sg.m9	sg.m2	PCIe 2 (x1)	PCIe 3 (x1)	1111
3358	sg.m9	sg.m2	PCIe 2 (x1)	SATA	2221
3558	sg.m9	PCIe 1 (x1)	PCIe 2 (x1)	SATA	2221
3555	sg.m9	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x1)	1111
7000	PCIe 1 (x4)				1111
9998	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x1)	SATA	2221

Table continues on the next page...

**Table 33-1. Supported SerDes options (continued)**

6058	PCIe 1 (x2)		PCIe 2 (x1)	SATA	2221
1455	xfi.m9	qs.m1,2,5,6	PCIe 2 (x1)	PCIe 3 (x1)	1222
2455	sg.m9 (2.5G)	qs.m1,2,5,6	PCIe 2 (x1)	PCIe 3 (x1)	1222
2255	sg.m9 (2.5G)	sg.m2 (2.5G)	PCIe 2 (x1)	PCIe 3 (x1)	1122
3333	sg.m9	sg.m2	sg.m5	sg.m6	1111
1460	xfi.m9	qs.m1,2,5,6	PCIe 3 (x2)		1222
2460	sg.m9 (2.5G)	qs.m1,2,5,6	PCIe 3 (x2)		1222
2560	sg.m9 (2.5G)	PCIe 1 (x1)	PCIe 3 (x2)		1222
3460	sg.m9	qs.m1,2,5,6	PCIe 3 (x2)		1222
3560	sg.m9	PCIe 1 (x1)	PCIe 3 (x2)		1111
3455	sg.m9	qs.m1,2,5,6	PCIe 2 (x1)	PCIe 3 (x1)	1222
9960	PCIe 1 (x1)	PCIe 2 (x1)	PCIe 3 (x2)		1111
2233	sg.m9 (2.5G)	sg.m2 (2.5)	sg.m5	sg.m6	1122
2533	sg.m9 (2.5G)	PCIe 1 (x1)	sg.m5	sg.m6	1222
2260	sg.m9 (2.5G)	sg.m2 (2.5G)	PCIe 3 (x2)		1122
	RESERVED				

### 33.1.1.2.1 SerDes lane assignments

The following table shows the SerDes lanes correspond to the SD1\_TXn\_P/N and SD1\_RXn\_P/N signals

**Table 33-2. Lane/Signal Assignments for SerDes**

Lane A	Lane B	Lane C	Lane D
SD1_RX0_P	SD1_RX1_P	SD1_RX2_P	SD1_RX3_P
SD1_RX0_N	SD1_RX1_N	SD1_RX2_N	SD1_RX3_N
SD1_TX0_P	SD1_TX1_P	SD1_TX2_P	SD1_TX3_P
SDI_TX0_N	SDI_TX1_N	SDI_TX2_N	SDI_TX3_N

### 33.1.1.2.2 Frame manager MACs

Each FMan supports seven MACs. These MAC's support different protocols as summarized in the table below.

**Table 33-3. MAC Capabilities**

MAC	RGMII 1 Gbps	SGMII 1 Gbps	SGMII 2.5 Gbps	XFI 10 Gbps
1	-	Y	-	-
2	-	Y	Y	-

*Table continues on the next page...*

**Table 33-3. MAC Capabilities (continued)**

MAC	RGMII 1 Gbps	SGMII 1 Gbps	SGMII 2.5 Gbps	XFI 10 Gbps
3	Y	-	-	-
4	Y	-	-	-
5	-	Y	-	-
6	-	Y	-	-
9	-	Y	Y	Y

Notes: RGMII Interfaces: MAC3 and MAC4 are used for EC1 and EC2 interfaces respectively.

### 33.1.1.2.3 Disabling unused SerDes modules

In order to disable the SerDes1 module, the following should be configured:

- SRDS\_PLL\_PD\_S1 = 2'b11 (both PLLs configured as powered down)
- SRDS\_PLL\_REF\_CLK\_SEL\_S1 = 2'b00
- SRDS\_PRTCL\_S1=0x0000 (no other values are permitted when both PLLs are powered down)

### 33.1.1.3 Powering down Unused serDes lanes

The powerdown of SerDes data lanes can be achieved by software using registers in the SerDes control block. Refer RCW[SRDS\_PLL\_PD\_S1] bit definition to see how lanes can be powered down based on the powering down of individual PLLs.

## 33.1.2 SerDes clocking

### 33.1.2.1 Valid reference clocks and PLL configurations for SerDes protocols

Each supported SerDes protocol allows for a finite set of valid SerDes-related RCW fields and reference clock frequencies, as shown in the following table:

**Table 33-4. Valid SerDes RCW Encodings and Reference Clocks**

SerDes protocol (given lane)	Valid reference clock frequency	Valid setting for SRDS_PRTCL_Sn	Valid setting for SRDS_PLL_REF _CLK_SEL_Sn		Valid setting for SRDS_DIV_*_Sn
			PLL1	PLL2	
PCI Express 2.5 Gbps (doesn't negotiate upwards)	100 MHz	Any PCIe	0: 100 MHz	0: 100 MHz	10: 2.5G
	125 MHz		1: 125 MHz	1: 125 MHz	
PCI Express 5 Gbps (can negotiate up to 5 Gbps)	100 MHz	Any PCIe	0: 100 MHz	0: 100 MHz	01: 5G
	125 MHz		1: 125 MHz	1: 125 MHz	
SATA (1.5, 3, 6 Gbps)	100 MHz	Any SATA	0: 100 MHz	-	Don't Care
	125 MHz		1: 125 MHz	-	
SGMII (1.25 Gbps)	100 MHz	SGMII @ 1.25 Gbps	0: 100 MHz	0: 100 MHz	Don't Care
	125 MHz		1: 125 MHz	1: 125 MHz	
2.5 G SGMII (3.125 Gbps)	125 Mhz	SGMII @ 3.125 Gbps	0: 125 MHz	-	Don't Care
	156.25 MHz		1: 156.25 MHz	-	
QSGMII (5 Gbps)	100 MHz	Any QSGMII	0: 100 MHz	0: 100 MHz	Don't Care
	125 MHz		1: 125 MHz	1: 125 MHz	
XFI (10.3125 Gbps)	156.25 Mhz		1: 156.25 MHz	-	-
				-	

Notes:

- 1) A spread-spectrum reference clock is permitted for PCI Express. However, if any other high speed interface such as SGMII, QSGMII, or SATA is used concurrently on the same SerDes bank, spread-spectrum clocking is not permitted. Only 0x7000 option supports spread-spectrum clock.
- 2) The clock for SATA, 2.5 G SGMII, and XFI can only be sourced from SerDes PLL1. Selecting a valid SerDes protocol (see [SerDes protocols](#)) automatically configures respective PLLs. However, a valid reference clock of either 100 MHz or 125 MHz is also required on reference input for PLL2 if the remaining SerDes lanes are used.

## 33.2 Overview

The SerDes module implements link serialization/deserialization and PCS functions for high speed serial interfaces from 1.25 Gbaud to 10.3125 Gbaud.

### 33.2.1 Features

The SerDes module includes 4 data lanes and 2 PLLs and supports the following protocols:

- PCI Express 3.0 (November, 2010) @ 2.5, 5 Gbaud
- SGMII ENG-46158, Revision 1.9 (July, 2009) @ 1.25, 3.125 Gbaud
- QSGMII EDCS-540123, Revision 1.2 (September, 2007) @ 5 Gbaud

## Modes of Operation

- IEEE 802.3-2015
  - 1000Base-KX @ 1.25 Gbaud
  - XFI @ 10.3125 Gbaud
- Serial ATA Revision 3.0 (June, 2009) @ 1.5, 3.0, 6.0 Gbaud

## 33.3 Modes of Operation

The SerDes supports several combinations of protocols and frequencies. See [SerDes lane assignments and multiplexing](#), for details on the combinations.

## 33.4 External Signals Description

The table below lists the external signals for the SerDes.

**Table 33-5. SerDes Interface Signals**

Pin Name	Description	No. of Signals	I/O
SDn_TX[0:3]_P	Transmitter serial output, positive data	4	O
SDn_TX_B[0:3]_N	Transmitter serial output, negative data	4	O
SDn_IMP_CAL_TX	Tx Impedance Calibration	1	I
SDn_RX[0:3]_P	Receiver serial output, positive data	4	I
SDn_RX_B[0:3]_N	Receiver serial output, negative data	4	I
SDn_IMP_CAL_RX	Rx Impedance Calibration	1	I
x=1, 2			
SDn_REFx_CLK_P	Reference clock input to PLLx	2	I
SDn_REFx_CLK_N	Reference clock-bar input to PLLx	2	I

The table below shows the mapping of external interface pins to register lane numbering.

**Table 33-6. Lane Mapping**

Lane	External Interface Pins
0	SD_TX0_P/SD_TX0_N + SD_RX0_P/SD_RX0_N
1	SD_TX1_P/SD_TX1_N + SD_RX1_P/SD_RX1_N
2	SD_TX2_P/SD_TX2_N + SD_RX2_P/SD_RX2_N
3	SD_TX3_P/SD_TX3_N + SD_RX3_P/SD_RX3_N

## 33.5 SerDes register descriptions

The SerDes module is programmed by control/status registers (CSRs). The CSRs are used for mode control, and to extract status information. All accesses to and from the registers must be made as 32-bit accesses. There is no support for accesses of sizes other than 32 bits. Writes to reserved register bits must always preserve the previous value; that is, the register should be programmed by reading the value, modifying appropriate fields, and writing back the value. Unless otherwise specified, the read value of unmapped registers or of reserved bits in mapped registers is not defined, and must not be assumed to be 0.

The table below lists the address, name, and a cross-reference to the complete description of each register. The offsets to the memory map table are defined for each SerDes module from 0x0000 to 0x1FFF (8 KB).

Unlisted register addresses are reserved.

- Reserved fields are always ignored for the purposes of determining access type.
- Reserved registers and fields must be preserved on writes.
- R (read only) indicates that all the non-reserved fields in a register are read-only.
- R/W (read/write) indicates all of the non-reserved fields in a register are either read/write or read-only, with at least one read/write field. The register description indicates which fields are read-write and which read-only.
- RC (read clear) indicates all of the fields in a register are cleared on read, and are not otherwise writeable.
- W1C indicates that all of the non-reserved fields in a register are cleared by writing ones to them.
- Mixed indicates a combination of access types.
- Special is used when no other category applies. In this case the register figure and field description provide the details for access.

This section describes the control, status, and test registers for SerDes and Protocol PCS layers.

All reserved register fields must be preserved on register writes (read-modified-write).

### 33.5.1 SerDes memory map

SerDes base address: 1EA\_0000h

## SerDes register descriptions

Offset	Register	Width (In bits)	Access	Reset value
0h	SerDes PLL1 Reset Control Register (PLL1RSTCTL)	32	RW	0467_452Fh
4h	SerDes PLL1 Control Register 0 (PLL1CR0)	32	RW	8080_0008h
8h	SerDes PLL1 Control Register 1 (PLL1CR1)	32	RW	0800_4100h
18h	SerDes PLL1 Control Register 5 (PLL1CR5)	32	RW	0000_0000h
20h	SerDes PLL2 Reset Control Register (PLL2RSTCTL)	32	RW	0467_452Fh
24h	SerDes PLL2 Control Register 0 (PLL2CR0)	32	RW	8080_0008h
28h	SerDes PLL2 Control Register 1 (PLL2CR1)	32	RW	0800_4100h
38h	SerDes PLL2 Control Register 5 (PLL2CR5)	32	RW	0000_0000h
90h	SerDes Transmit Calibration Control Register (TCALCR)	32	RW	0000_0000h
94h	SerDes Transmit Calibration Control Register 1 (TCALCR1)	32	RW	0000_0000h
A0h	Receive Calibration Control Register (RCALCR)	32	RW	0000_0000h
A4h	SerDes Receive Calibration Control Register 1 (RCALCR1)	32	RW	0000_0000h
B0h	General Control Register 0 (GR0)	32	RW	0000_0000h
100h	Lane A Protocol Select Status Register 0 (LNAPSSR0)	32	RO	0000_0000h
120h	Lane B Protocol Select Status Register 0 (LNPSSR0)	32	RO	0000_0000h
140h	Lane C Protocol Select Status Register 0 (LNCPSSR0)	32	RO	0000_0000h
160h	Lane D Protocol Select Status Register 0 (LNDPSSR0)	32	RO	0000_0000h
200h	Protocol Configuration Register 0 (PCCR0)	32	RW	0000_0000h
208h	Protocol Configuration Register 2 (PCCR2)	32	RW	0000_0000h
220h	Protocol Configuration Register 8 (PCCR8)	32	RW	0000_0000h
224h	Protocol Configuration Register 9 (PCCR9)	32	RW	0000_0000h
22Ch	Protocol Configuration Register B (PCCRB)	32	RW	0000_0000h
800h	General Control Register 0 - Lane A (LNAGCR0)	32	RW	1104_0000h
804h	General Control Register 1 - Lane A (LNAGCR1)	32	RW	004C_4011h
80Ch	Speed Switch Control Register 0 - Lane A (LNASSCR0)	32	RW	9828_2B00h
810h	Receive Equalization Control Register 0 - Lane A (LNARECR0)	32	RW	0000_001Fh
814h	Receive Equalization Control Register 1 - Lane A (LNARECR1)	32	RO	0000_0008h
818h	Transmit Equalization Control Register 0 - Lane A (LNATECR0)	32	RW	1028_3000h
81Ch	Speed Switch Control Register 1 - Lane 0 (LNASSCR1)	32	RW	0000_0000h
820h	TTL Control Register 0 - Lane A (LNATTLCR0)	32	RW	0000_0400h
83Ch	Test Control/Status Register 3 - Lane A (LNATCSR3)	32	RW	0400_0000h
840h	General Control Register 0 - Lane B (LNBGCR0)	32	RW	1104_0000h
844h	General Control Register 1 - Lane B (LNBGCR1)	32	RW	004C_4011h
84Ch	Speed Switch Control Register 0 - Lane B (LNBSSCR0)	32	RW	9828_2B00h
850h	Receive Equalization Control Register 0 - Lane B (LNBRECR0)	32	RW	0000_001Fh
854h	Receive Equalization Control Register 1 - Lane B (LNBRECR1)	32	RO	0000_0008h
858h	Transmit Equalization Control Register 0 - Lane B (LNBTECR0)	32	RW	1028_3000h
85Ch	Speed Switch Control Register 1 - Lane 0 (LNBSSCR1)	32	RW	0000_0000h
860h	TTL Control Register 0 - Lane B (LNBTTLCR0)	32	RW	0000_0400h
87Ch	Test Control/Status Register 3 - Lane B (LNBTCR3)	32	RW	0400_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
880h	General Control Register 0 - Lane C (LNCGCR0)	32	RW	1104_0000h
884h	General Control Register 1 - Lane C (LNCGCR1)	32	RW	004C_4011h
88Ch	Speed Switch Control Register 0 - Lane C (LNCSSCR0)	32	RW	9828_2B00h
890h	Receive Equalization Control Register 0 - Lane C (LNCRECR0)	32	RW	0000_001Fh
894h	Receive Equalization Control Register 1- Lane C (LNCRECR1)	32	RO	0000_0008h
898h	Transmit Equalization Control Register 0 - Lane C (LNCTECR0)	32	RW	1028_3000h
89Ch	Speed Switch Control Register 1- Lane 0 (LNCSSCR1)	32	RW	0000_0000h
8A0h	TTL Control Register 0 - Lane C (LNCTTLCR0)	32	RW	0000_0400h
8BCh	Test Control/Status Register 3 - Lane C (LNCTCSR3)	32	RW	0400_0000h
8C0h	General Control Register 0 - Lane D (LNDGCR0)	32	RW	1104_0000h
8C4h	General Control Register 1 - Lane D (LNDGCR1)	32	RW	004C_4011h
8CCh	Speed Switch Control Register 0 - Lane D (LNDSSCR0)	32	RW	9828_2B00h
8D0h	Receive Equalization Control Register 0 - Lane D (LNDRECR0)	32	RW	0000_001Fh
8D4h	Receive Equalization Control Register 1- Lane D (LNDRECR1)	32	RO	0000_0008h
8D8h	Transmit Equalization Control Register 0 - Lane D (LNDTECR0)	32	RW	1028_3000h
8DCh	Speed Switch Control Register 1- Lane 0 (LNDSSCR1)	32	RW	0000_0000h
8E0h	TTL Control Register 0 - Lane D (LNDTTLCR0)	32	RW	0000_0400h
8FCh	Test Control/Status Register 3 - Lane D (LNCTCSR3)	32	RW	0400_0000h
1000h	PEXA Protocol Control Register 0 (PEXACR0)	32	RW	2000_0000h
1040h	PEXB Protocol Control Register 0 (PEXBCR0)	32	RW	2000_0000h
1080h	PEXC Protocol Control Register 0 (PEXCCR0)	32	RW	2000_0000h
1804h	SGMIIA Protocol Control Register 1 (SGMIIACR1)	32	RW	0000_0000h
180Ch	SGMIIA Protocol Control Register 3 (SGMIIACR3)	32	RO	0000_0000h
1814h	SGMIIIB Protocol Control Register 1 (SGMIIIBCR1)	32	RW	0000_0000h
181Ch	SGMIIIB Protocol Control Register 3 (SGMIIIBCR3)	32	RO	0000_0000h
1824h	SGMIIC Protocol Control Register 1 (SGMIICCR1)	32	RW	0000_0000h
182Ch	SGMIIC Protocol Control Register 3 (SGMIICCR3)	32	RO	0000_0000h
1834h	SGMIID Protocol Control Register 1 (SGMIIDCR1)	32	RW	0000_0000h
183Ch	SGMIID Protocol Control Register 3 (SGMIIDCR3)	32	RO	0000_0000h
1884h	QSGMIIA Protocol Control Register 1 (QSGMIIACR1)	32	RW	0000_0000h
188Ch	QSGMIIA Protocol Control Register 3 (QSGMIIACR3)	32	RO	0000_0000h
1894h	QSGMIIIB Protocol Control Register 1 (QSGMIIIBCR1)	32	RW	0000_0000h
189Ch	QSGMIIIB Protocol Control Register 3 (QSGMIIIBCR3)	32	RO	0000_0000h
1984h	XFI A Protocol Control Register 1 (XFIACR1)	32	RW	0000_0000h
198Ch	XFI A Protocol Control Register 3 (XFIACR3)	32	RO	0000_0000h
1994h	XFI B Protocol Control Register 1 (XFIBCR1)	32	RW	0000_0000h
199Ch	XFI B Protocol Control Register 3 (XFIBCR3)	32	RO	0000_0000h

## 33.5.2 SerDes PLLa Reset Control Register (PLL1RSTCTL - PLL2RSTCTL)

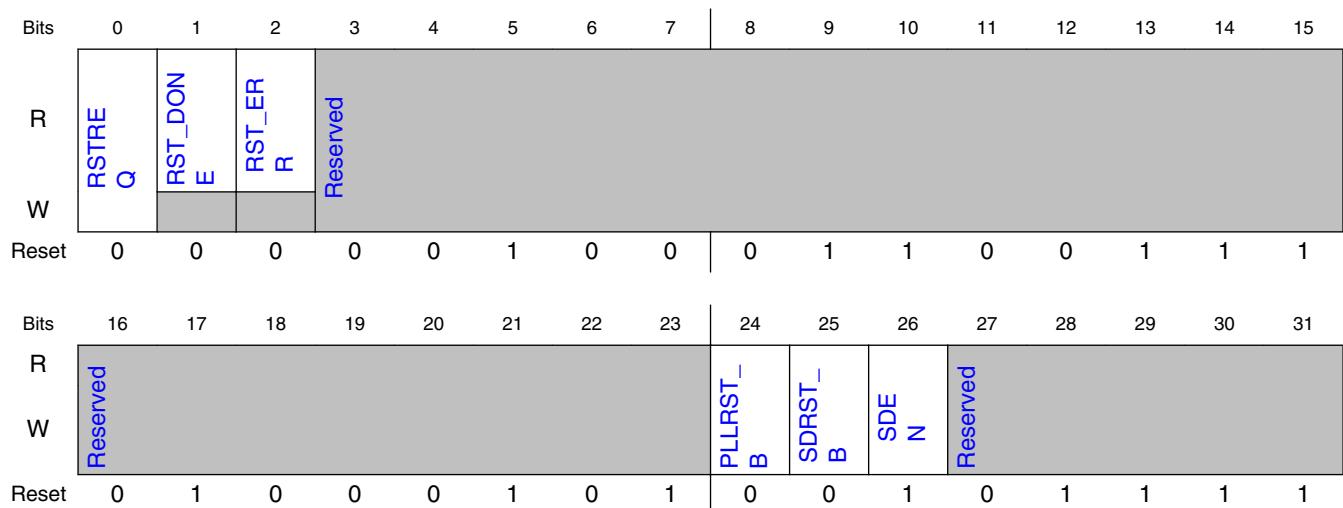
### 33.5.2.1 Offset

Register	Offset
PLL1RSTCTL	0h
PLL2RSTCTL	20h

### 33.5.2.2 Function

PLL $n$ RSTCTL contains control and status bits for the SerDes PLL  $n$  reset state machine. This register should not be modified unless RST\_ERR=1 or RST\_DONE=1

### 33.5.2.3 Diagram



### 33.5.2.4 Fields

Field	Function
0 RSTREQ	Reset request PLL Reset Request - write 1 self-clearing. Software setting of 1 initiates a soft reset of SerDes PLL $n$ , along with all lanes using PLL $n$ .

Table continues on the next page...

Field	Function
	<p>Hardware automatically clears this bit before reset is done.</p> <p>Note: This field must be 0 if PLL2 does not have a reference clock. Clearing this bit during the reset sequence has no effect.</p> <p>0b - Not requesting PLL soft reset. 1b - Requesting PLL soft reset.</p>
1 RST_DONE	<p>Reset done</p> <p>PLL Reset Done from Control Block State Machine</p> <p>0b - PLL reset sequence in progress. 1b - PLL reset sequence complete.</p>
2 RST_ERR	<p>Reset Error</p> <p>No PLL Lock before counter time_out</p> <p>0b - Normal function 1b - PLL lock didn't happen in the expected time period</p>
3-23 —	Reserved
24 PLLRST_B	<p>PLL reset. Resets PLLn</p> <p>PLL reset. Resets PLLn .</p> <p>Default value: 0</p> <p>The SerDes reset state machine overrides this field to 0 in some states. Setting PLLRST_B=0 when RST_DONE=0 forces a SerDes PLL reset regardless of the reset state machine state, and may cause the reset state machine to time out and set RST_ERR=1.</p> <p>0b - Force PLL reset 1b - Application Mode, unless RSTREQ=1</p>
25 SDRST_B	<p>SRDS group reset. Resets all lanes operating from PLLn</p> <p>The SerDes reset state machine overrides this field to 0 in some states.</p> <p>0b - Force SerDes group reset 1b - Application Mode, unless RSTREQ=1</p>
26 SDEN	<p>SerDes enable. Enable the section of the SerDes module clocked by PLLn</p> <p>SerDes enable. Enable the section of the SerDes module clocked by PLLn.</p> <p>Default value: 1</p> <p>The SerDes reset state machine overrides this field to 0 in some states. Setting SDEN=0 when RST_DONE=0 forces a SerDes powerdown regardless of the reset state machine state, and may cause the reset state machine to time out and set RST_ERR=1.</p> <p>Note: This field must be 0 if PLL2 does not have a reference clock.</p> <p>0b - Force SerDes power down 1b - Application Mode, unless RSTREQ=1</p>
27-31 —	Reserved

### 33.5.3 SerDes PLLa Control Register 0 (PLL1CR0 - PLL2CR0)

#### 33.5.3.1 Offset

Register	Offset
PLL1CR0	4h
PLL2CR0	24h

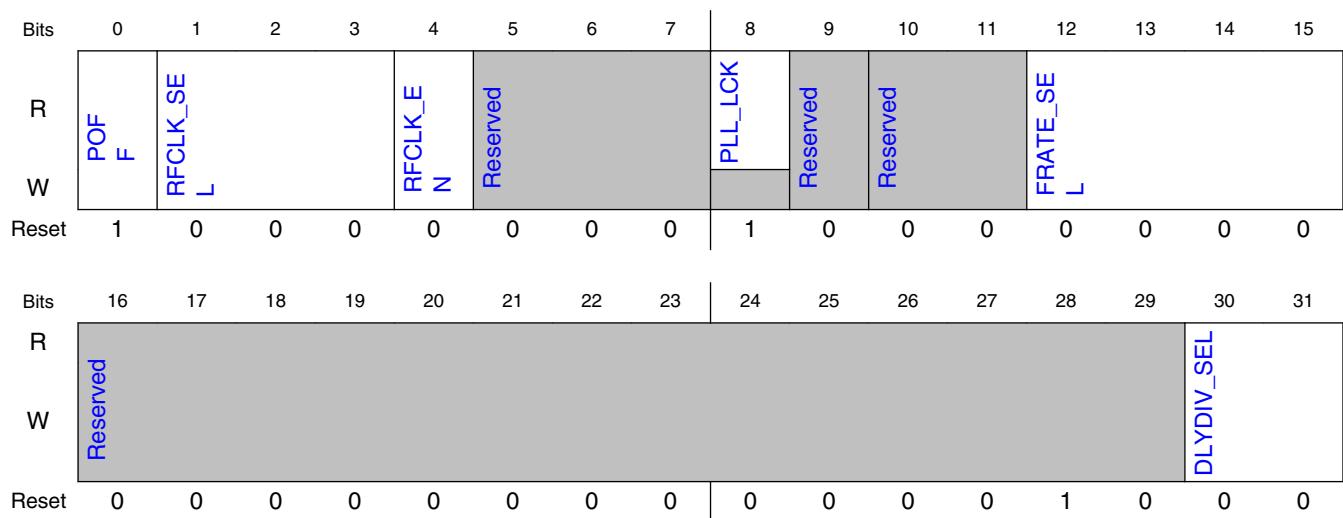
#### 33.5.3.2 Function

PLL<sub>n</sub>CR0 contains control and status bits for SerDes PLL *n*

##### NOTE

This register may be automatically updated when PCI Express is active on the SerDes. Only write to this register when all associated PCI Express controllers are inactive.

#### 33.5.3.3 Diagram



#### 33.5.3.4 Fields

Field	Function
0	PLL off

Table continues on the next page...

Field	Function
P0FF	<p>Power down an unused PLL</p> <p>Default value: 0</p> <p>0b - PLL on 1b - PLL off. All lanes referencing this PLL must also be disabled.</p>
1-3 RFCLK_SEL	<p>Reference clock frequency select</p> <p>All others reserved</p> <p>Default value set by reset configuration. Recommended setting per protocol:</p> <ul style="list-style-type: none"> <li>• SATA: 000, 001 or 011</li> <li>• PCIe: 000 or 001</li> <li>• SGMII @ 3.125: 001 or 010</li> <li>• SGMII/1000Base-KX @ 1.25: 000 or 001</li> <li>• QSGMII: 000, 001 or 010</li> <li>• XFI: 010 or 100</li> </ul> <p>000b - 100 MHz 001b - 125 MHz 010b - 156.25 MHz 011b - 150 MHz</p>
4 RFCLK_EN	<p>Reference clock observe enable</p> <p>Default value: 0</p> <p>Recommended setting: 0</p> <p>0b - Disable reference clock output 1b - Enable buffered version of SD_REF_CLKn</p>
5-7 —	Reserved
8 PLL_LCK	<p>PLL lock</p> <p>Indicates PLL(n) has calibrated and locked</p> <p>0b - PLL is not locked 1b - PLL is locked</p>
9 —	Reserved
10-11 —	Reserved
12-15 FRATE_SEL	<p>Select frequency of PLL VCO. All lanes within a group must operate at a multiple of this frequency and within specified limits of the protocol(s).</p> <p>Default value set by reset configuration. Required settings per protocol:</p> <p>All others are reserved</p> <p>0000b - 5.00 GHz ----- [ SGMII QSGMII PCIe ] 0110b - 5.15625 GHz ---- [ XFI ] 0111b - 4.00 GHz ----- 1001b - 3.125 GHz ----- [ SGMII ] 1010b - 3.00 GHz ----- [ SATA ]</p>
16-29 —	Reserved
30-31	Select PLLn_ex_dly_clk divider value

## SerDes register descriptions

Field	Function
DLYDIV_SEL	<p>This feature is used for:</p> <ul style="list-style-type: none"> <li>• 1000Base-KX</li> </ul> <p>When that mode is not active, DLYDIV_SEL should be 00.</p> <p>00b - PLLn_ex_dly_clk off 01b - PLLn_ex_dly_clk = FRATE_SEL/16 (e.g. 312.5 MHz for 5.0 GHz)</p>

## 33.5.4 SerDes PLLa Control Register 1 (PLL1CR1 - PLL2CR1)

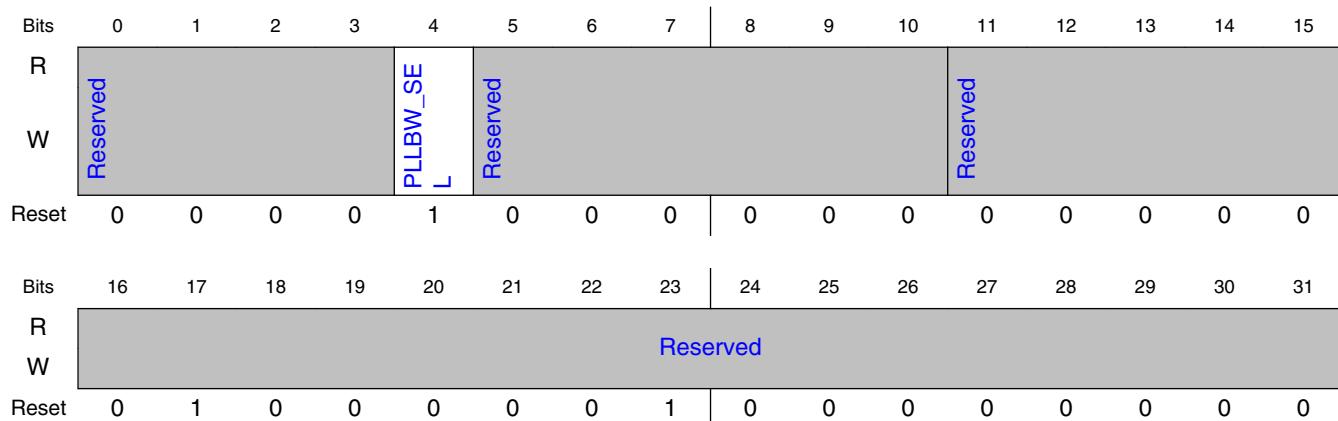
### 33.5.4.1 Offset

Register	Offset
PLL1CR1	8h
PLL2CR1	28h

### 33.5.4.2 Function

PLLnCR1 contains control bits for SerDes PLL *n*.

### 33.5.4.3 Diagram



### 33.5.4.4 Fields

Field	Function
0-3	Reserved
—	
4 PLLBW_SEL	Select Higher PLL(n) Bandwidth Recommended setting per PLL: 1 0b - Nominal PLL Bandwidth 1b - PLL Bandwidth Setting
5-10	Reserved
—	
11-31	Reserved
—	

## 33.5.5 SerDes PLLa Control Register 5 (PLL1CR5 - PLL2CR5)

### 33.5.5.1 Offset

Register	Offset
PLL1CR5	18h
PLL2CR5	38h

### 33.5.5.2 Function

PLLnCR5 contains control bits for the SerDes x PLL *n*.

### 33.5.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<code>SEL_REFCLK_AMP_DIS</code>	Reserved			<code>LEFT_REF_BUF_EN</code>	Reserved		<code>RIGHT_REF_BUF_EN</code>	Reserved							
W	S	N			N			N								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.5.4 Fields

Field	Function
0 <code>SEL_REFCLK_AMP_DIS</code>	Disables the reference clock amplifier and selects usage of the reference clock coming from the left, either from SoC or from another 10G Serdes PLL Default value set by reset
1-3 —	Reserved
4 <code>LEFT_REF_BUF_EN</code>	Enable for left directed reference clock buffer
5-6 —	Reserved
7 <code>RIGHT_REF_BUF_EN</code>	Enable for right directed reference clock buffer
8-31 —	Reserved

## 33.5.6 SerDes Transmit Calibration Control Register (TCALCR)

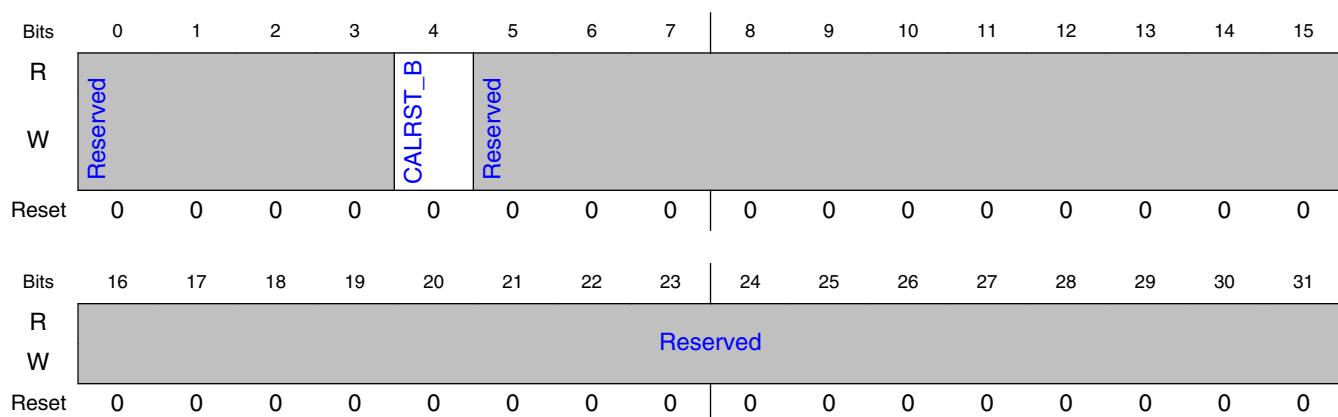
### 33.5.6.1 Offset

Register	Offset
TCALCR	90h

### 33.5.6.2 Function

TCALCR contains the control bits used for Transmit Calibration.

### 33.5.6.3 Diagram



### 33.5.6.4 Fields

Field	Function
0-3 —	Reserved
4 CALRST_B	Reset the transmit calibration During POR (warm reset sequence) it is active and is released when the first PLL comes out of reset. 0b - Reset 1b - Application Mode
5-31 —	Reserved

## 33.5.7 SerDes Transmit Calibration Control Register 1 (TCAL CR1)

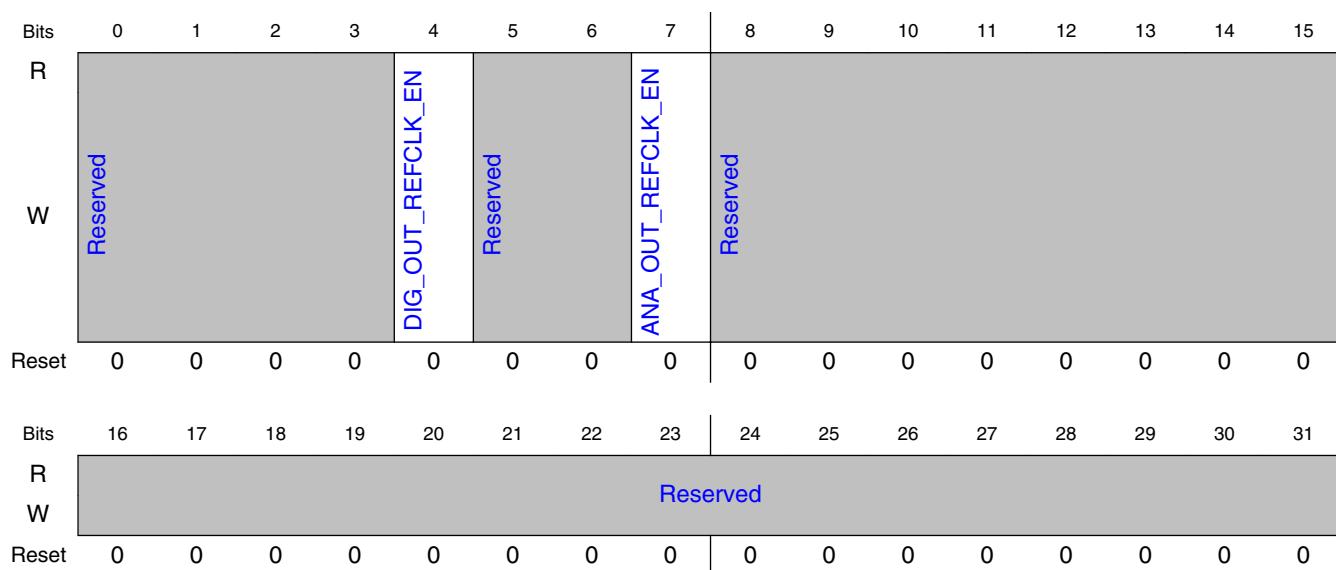
### 33.5.7.1 Offset

Register	Offset
TCALCR1	94h

### 33.5.7.2 Function

TCALCR1 contains the bits used for reference clock controls in the right half-lane.

### 33.5.7.3 Diagram



### 33.5.7.4 Fields

Field	Function
0-3	Reserved
—	

Table continues on the next page...

Field	Function
4 DIG_OUT_REF CLK_EN	Enable CMOS digital buffered version of selected ref_clk to the right. Used in conjunction with PLLn_CR5[RIGHT_REF_BUF_EN]
5-6 —	Reserved
7 ANA_OUT_REF CLK_EN	Enable CML analog buffered version of selected ref_clk to the left. Used in conjunction with PLLn_CR5[RIGHT_REF_BUF_EN].
8-31 —	Reserved

## 33.5.8 Receive Calibration Control Register (RCALCR)

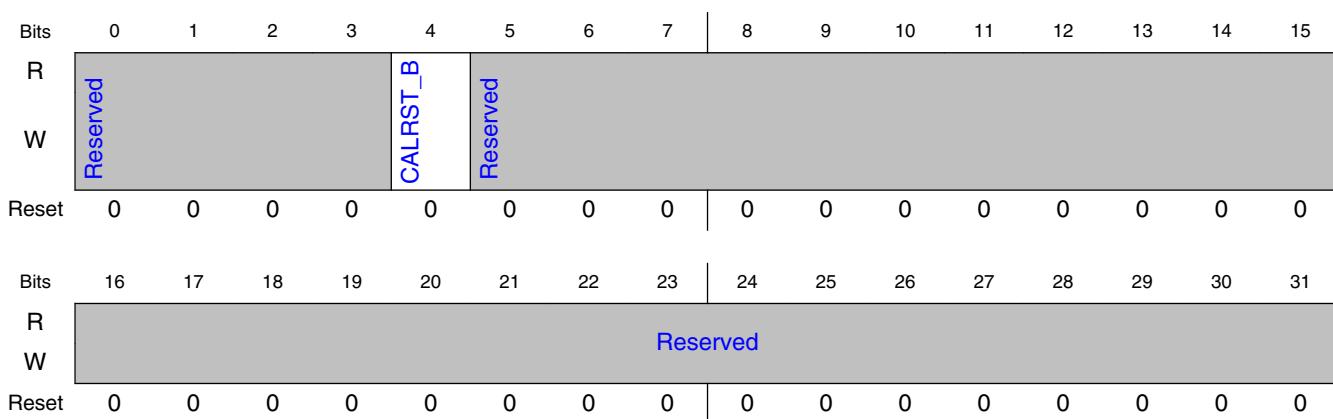
### 33.5.8.1 Offset

Register	Offset
RCALCR	A0h

### 33.5.8.2 Function

RCALCR contains the control bits for Receive Calibration.

### 33.5.8.3 Diagram



### 33.5.8.4 Fields

Field	Function
0-3 —	Reserved
4 CALRST_B	Reset the receiver calibration During POR (warm reset sequence) it is active and is released when the first PLL comes out of reset 0b - Reset 1b - Application Mode
5-31 —	Reserved

## 33.5.9 SerDes Receive Calibration Control Register 1 (RCALCR1)

### 33.5.9.1 Offset

Register	Offset
RCALCR1	A4h

### 33.5.9.2 Function

RCALCR1 contains the bits for reference clock controls on the left half-lane.

### 33.5.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<code>ANA_IN_REFCLK_BUF_EN</code>	Reserved			<code>DIG_OUT_REFCLK_EN</code>	Reserved		<code>ANA_OUT_REFCLK_EN</code>	Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.9.4 Fields

Field	Function
0 ANA_IN_REFCLK_BUF_EN	Enable CML analog buffer used to transition SoC reference clock to 10G Serdes
1-3 —	Reserved
4 DIG_OUT_REFCLK_EN	Enable CMOS digital buffered version of selected ref_clk to the left. Used in conjunction with PLLn_CR5[LEFT_REF_BUF_EN].
5-6 —	Reserved
7 ANA_OUT_REFCLK_EN	Enable CML analog buffered version of selected ref_clk to the left. Used in conjunction with PLLn_CR5[LEFT_REF_BUF_EN].
8-29 —	Reserved

Table continues on the next page...

## SerDes register descriptions

Field	Function
30-31 BI_REFCTL_OUT	Output COP DFT/BurnIn/JTAG reference clock controls to wrapper

## 33.5.10 General Control Register 0 (GR0)

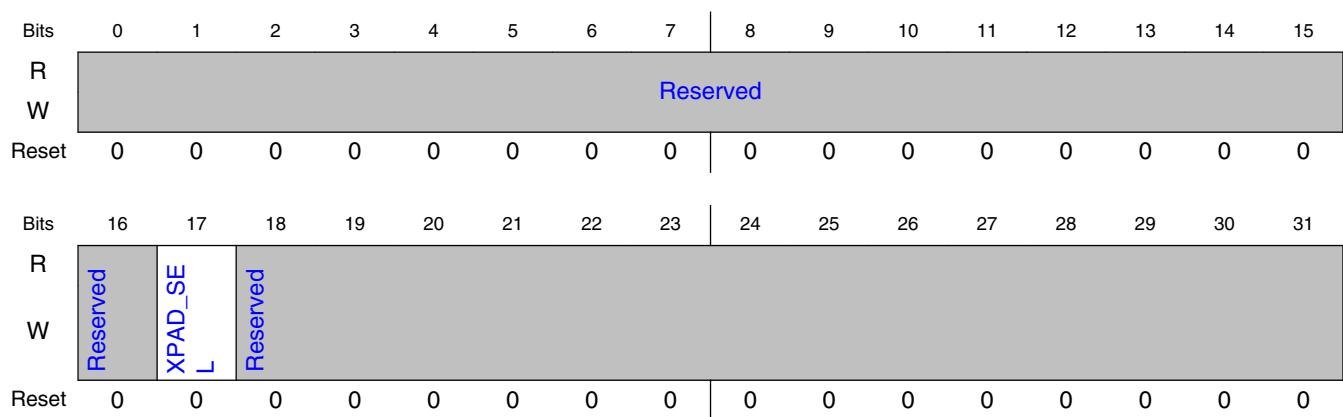
### 33.5.10.1 Offset

Register	Offset
GR0	B0h

### 33.5.10.2 Function

GR0 contains general control/status bits for the SerDes.

### 33.5.10.3 Diagram



### 33.5.10.4 Fields

Field	Function
0-16	Reserved

*Table continues on the next page...*

Field	Function
—	
17 XPAD_SEL	<p>Describes to SerDes module the value of the power supply being used by the Serdes I/Os: Full SerDes reset required after changing this value.</p> <p><b>Warning:</b> setting XPAD_SEL to low xpadvdd supply when the supply is &gt; 1.35V+5% may cause irreparable damage to the device.</p> <p>0b - High xpadvdd supply (nominal 1.5V - see H/W spec for details) 1b - Low xpadvdd supply (nominal 1.35V - see H/W spec for details)</p>
18-31	Reserved
—	

### 33.5.11 Lane a Protocol Select Status Register 0 (LNAPSSR0 - LNDPSSR0)

#### 33.5.11.1 Offset

Register	Offset
LNAPSSR0	100h
LNPSSR0	120h
LNCPSSR0	140h
LNDPSSR0	160h

#### 33.5.11.2 Function

LNmPSSR0 contains decoded protocol select information per lane.

The primary usage of this register is to map an Ethernet link to its corresponding MAC and MDIO address space.

### 33.5.11.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	TYP	E						Reserved				MAC			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R			Reserved			PCS			Reserved				LANE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.11.4 Fields

Field	Function
0 —	Reserved
1-7 TYPE	Protocol Type Bits 1-5 are same as LNMGCR0[PROTS]. Bits 6-7: <ul style="list-style-type: none"><li>• QSGMII: 01</li><li>• All others: 00</li></ul>
8-11 —	Reserved
12-15 MAC	MAC instance Note: if TYPE[6:7]=QSGMII, then the MAC instance listed is the MAC connected to port 0, which is also the MAC used for MDIO accesses.  0000b - MAC1 0001b - MAC2 0010b - MAC3 0011b - MAC4 0100b - MAC5 0101b - MAC6 0110b - MAC7 0111b - MAC8 1000b - MAC9
16-20 —	Reserved
21-23 PCS	PCS instance of TYPE within PHY 000b - PCSa/1 001b - PCSb/2

Table continues on the next page...

Field	Function
	010b - PCSc/3 011b - PCSd/4
24-27 —	Reserved
28-31 LANE	Lane number within PCS  Example 1: for x4 PCIe [3:0] on lanes [3:0], LANE=0 for n=0, LANE=1 for n=1, LANE=2 for n=2 and LANE=3 for n=3  Note that the lane number within PCS does not take into account lane reversal, either via auto-negotiate, or by S/W control in SerDes registers.  All others reserved  0000b - lane 0 0001b - lane 1 0010b - lane 2 0011b - lane 3

### 33.5.12 Protocol Configuration Register 0 (PCCR0)

#### 33.5.12.1 Offset

Register	Offset
PCCR0	200h

#### 33.5.12.2 Function

PCCR0 contains the protocol configuration for PCIe.

### 33.5.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	PEXA_CFG			Reserved	PEXB_CFG			Reserved	PEXC_CFG			Reserved	Reserved		
W		G				G				G						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.12.4 Fields

Field	Function
0 —	Reserved
1-3 PEXA_CFG	PEXa Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 1 010b - x4 [3:0] on Lanes [0:3] 011b - x1 on Lane 0 100b - x2 [1:0] on Lanes [0:1]
4 —	Reserved
5-7 PEXB_CFG	PEXb Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 2 010b - x1 on Lane 1
8 —	Reserved
9-11 PEXC_CFG	PEXc Configuration All others reserved Default value set by RCW configuration. 000b - disabled

Table continues on the next page...

Field	Function
	001b - x1 on Lane 3 011b - x2 [1:0] on Lanes [2:3]
12 —	Reserved
13-15 —	Reserved
16-31 —	Reserved

### 33.5.13 Protocol Configuration Register 2 (PCCR2)

#### 33.5.13.1 Offset

Register	Offset
PCCR2	208h

#### 33.5.13.2 Function

PCCR2 contains the protocol configuration for SATA and Aurora.

#### 33.5.13.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	SATAA_CFG			Reserved	Reserved										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	Reserved			Reserved									Reserved	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.13.4 Fields

Field	Function
0 —	Reserved
1-3 SATAA_CFG	SATAa Configuration All others reserved 000b - Disabled 001b - Enabled on Lane 3
4 —	Reserved
5-7 —	Reserved
8-16 —	Reserved
17-19 —	Reserved
20-27 —	Reserved
28 —	Reserved
29-31 —	Reserved

### 33.5.14 Protocol Configuration Register 8 (PCCR8)

#### 33.5.14.1 Offset

Register	Offset
PCCR8	220h

#### 33.5.14.2 Function

PCCR8 contains the protocol configuration for SGMII /1000Base-KX.

### 33.5.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	SGMIIA_KX	SGMIIA_CFG			SGMIIB_KX	SGMIIB_CFG			SGMIIC_KX	SGMIIC_CFG			SGMIID_KX	SGMIID_CFG		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 33.5.14.4 Fields

Field	Function
0 SGMIIA_KX	SGMIIb 1000Base-KX Configuration: Note: this configuration bit must be set before performing any MDIO initialization of the PCS. 0b - SGMII mode 1b - 1000Base-KX mode
1-3 SGMIIA_CFG	SGMIIa Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 0 to FMan Mac 9
4 SGMIIB_KX	SGMIIC 1000Base-KX Configuration: Note: this configuration bit must be set before performing any MDIO initialization of the PCS. 0b - SGMII mode 1b - 1000Base-KX mode
5-7 SGMIIB_CFG	SGMIIb Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 2 to FMan Mac 2
8 SGMIIC_KX	SGMIId 1000Base-KX Configuration: Note: this configuration bit must be set before performing any MDIO initialization of the PCS. 0b - SGMII mode

Table continues on the next page...

## SerDes register descriptions

Field	Function
	1b - 1000Base-KX mode
9-11 SGMIIC_CFG	SGMIIC Configuration All others reserved Default value set by RCW configuration 000b - disabled 001b - x1 on Lane 3 to FMan Mac 5
12 SGMIIID_KX	SGMIIle 1000Base-KX Configuration: Note: this configuration bit must be set before performing any MDIO initialization of the PCS. 0b - SGMII mode 1b - 1000Base-KX mode
13-15 SGMIIID_CFG	SGMIIId Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 3 to FMan Mac 6
16 —	Reserved
17-19 —	Reserved
20 —	Reserved
21-23 —	Reserved
24 —	Reserved
25-27 —	Reserved
28 —	Reserved
29-31 —	Reserved

### 33.5.15 Protocol Configuration Register 9 (PCCR9)

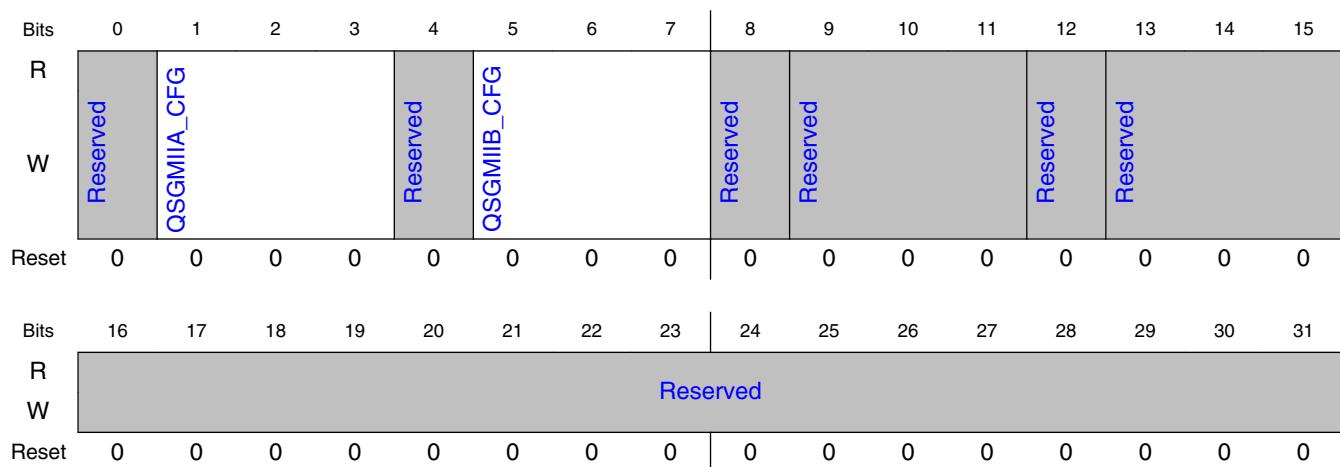
### 33.5.15.1 Offset

Register	Offset
PCCR9	224h

### 33.5.15.2 Function

PCCR9 contains the Protocol Configuration for QSGMII.

### 33.5.15.3 Diagram



### 33.5.15.4 Fields

Field	Function
0	Reserved
—	
1-3 QSGMIIA_CFG	QSGMIIa Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 0 connected to FMan MACs 1,2,5,6
4	Reserved
—	

Table continues on the next page...

## SerDes register descriptions

Field	Function
5-7 QSGMIIb_CFG	QSGMIIb Configuration All others reserved Default value set by RCW configuration. 000b - disabled 001b - x1 on Lane 1 connected to FMan MACs 1,2,5,6
8 —	Reserved
9-11 —	Reserved
12 —	Reserved
13-15 —	Reserved
16-31 —	Reserved

## 33.5.16 Protocol Configuration Register B (PCCRB)

### 33.5.16.1 Offset

Register	Offset
PCCRB	22Ch

### 33.5.16.2 Function

PCCRB contains the protocol configuration for XFI .

### 33.5.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved	XFIA_CFG		Reserved	XFIB_CFG		Reserved	Reserved		Reserved		Reserved	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved	Reserved		Reserved	Reserved		Reserved	Reserved	Reserved	Reserved		Reserved	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.16.4 Fields

Field	Function
0 —	Reserved
1-3 XFIA_CFG	XFIa Configuration All others reserved Default value set by RCW configuration. 000b - Disabled 001b - x1 on Lane 0 to FMan MAC 9
4 —	Reserved
5-7 XFIB_CFG	XFIb Configuration All others reserved Default value set by reset configuration. 000b - Disabled
8 —	Reserved
9-11 —	Reserved
12 —	Reserved
13-15 —	Reserved
16	Reserved

Table continues on the next page...

## SerDes register descriptions

Field	Function
—	
17-19	Reserved
—	
20	Reserved
—	
21-23	Reserved
—	
24	Reserved
—	
25-27	Reserved
—	
28	Reserved
—	
29-31	Reserved
—	

### 33.5.17 General Control Register 0 - Lane a (LNAGCR0 - LNDGCR0)

#### 33.5.17.1 Offset

Register	Offset
LNAGCR0	800h
LNBGCR0	840h
LNCGCR0	880h
LNDGCR0	8C0h

#### 33.5.17.2 Function

LNmGCR0 contains functional control bits for SerDes lane *m*.

Special consideration must be taken when writing this register while PLLnRSTCTL[RST\_DONE]=0, or if PCIe is running on this lane. See RPLL\_LES, TPLL\_LES, RRST\_B and TRST\_B fields for details.

See , for details on the sequence required to change settings.

### 33.5.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RPLL_L S	Reserved	RRAT_SE L		TPLL_L S	Reserved	TRAT_SE L		Reserved	RRST_B	TRST_B	RX_P D	TX_PD	IF20BIT_EN	Reserved	FIRST_LAN E
W	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0
Reset	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved				PROTS				Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.17.4 Fields

Field	Function
0 RPLL_LES	<p>Directs the RX portion of lane a to use the corresponding PLL.</p> <p>Default value set by reset configuration.</p> <p>Note: must be set the same as TPLL_LES for normal function.</p> <p>0b - use PLL2 1b - use PLL1</p>
1 —	Reserved
2-3 RRAT_SEL	<p>Receiver speed selection for lane a, relative to the corresponding PLLnCR0[FRATE_SEL], n as selected by LNaGCR0[RPLL_LES]:</p> <p>Default value set by reset configuration.</p> <p>Required setting per protocol:</p> <ul style="list-style-type: none"> <li>SGMII 10 (1.25 Gbaud)</li> <li>SGMII 00 (3.125 Gbaud)</li> <li>QSGMII 00</li> <li>XFI 11</li> <li>PCIe 01</li> <li>SATA 10</li> </ul> <p>This register field is read-only when running in PClear SATA mode. The above required values are defaults for gen1 initialization.</p> <p>Note: must be set the same as TRAT_SEL for normal function.</p> <p>00b - Same as FRATE_SEL 01b - FRATE_SEL/2 10b - FRATE_SEL/4 11b - FRATE_SEL*2</p>

Table continues on the next page...

## SerDes register descriptions

Field	Function
4 TPLLLES	<p>Used to direct the TX portion of lane to use the corresponding PLL.</p> <p>Default value set by reset configuration.</p> <p>Note: must be set the same as RPLLLES for normal function.</p> <p>See for details on the sequence required to change this setting.</p> <p>0b - use PLL2 1b - use PLL1</p>
5 —	Reserved
6-7 TRATSEL	<p>Transmitter speed selection for lane a, relative to the corresponding PLLnCR0[FRATE_SEL], n as selected by LNaGCR0[TPLLLES]:</p> <p>Default value set by reset configuration.</p> <p>Required setting per protocol: see RRATSEL</p> <p>This register field is read-only when running in PClear SATA mode.</p> <p>Note: must be set the same as RRATSEL for normal function.</p> <p>00b - Same as FRATE_SEL 01b - FRATE_SEL/2 10b - FRATE_SEL/4 11b - FRATE_SEL*2</p>
8 —	Reserved
9 RRSTB	<p>Resets receiver for lane a</p> <p>Recommended setting per protocol: 1, unless PLLnRSTCTL[RST_DONE]=0, then 0.</p> <p>This register value is read-only during SerDes reset (POR or PLLnRSTCTL[RSTREQ]).</p> <p>This register field is read-only when running in PClear SATA mode. In SATA mode, this field may read as 1 while a controller-driven lane reset is in progress.</p> <p>Used in lane reset and reconfiguring procedures. See <a href="#">Lane Reset and Reconfiguration</a>, for details.</p> <p>0b - Reset 1b - Application Mode</p>
10 TRSTB	<p>Resets transmitter for lane a</p> <p>Recommended setting per protocol: 1</p> <p>This register value is read-only during SerDes reset (POR or PLLnRSTCTL[RSTREQ]).</p> <p>This register field is read-only when running in PClear SATA mode.</p> <p>Used in lane reset and reconfiguring procedures. See <a href="#">Lane Reset and Reconfiguration</a>, for details.</p> <p>0b - Reset 1b - Application Mode</p>
11 RXPD	<p>Lane powerdown for receiver on lane a</p> <p>Default value set by reset configuration.</p> <p>This register field read-only during SerDes reset (POR or PLLnRSTCTL[RSTREQ]).</p> <p>See <a href="#">Lane Enable After Powerdown</a> for details on re-enabling powered down lanes.</p> <p>0b - Lane receiver active 1b - Lane receiver powered down</p>
12	Lane powerdown for transmitter on lane a

*Table continues on the next page...*

Field	Function
TX_PD	<p>Default value set by reset configuration.</p> <p>This register value is read-only during SerDes reset (POR or PLLnRSTCTL[RSTREQ]).</p> <p>Note: the user must not assert TX_PD for the master clock lane (TX_CLK_1STLANE=1) of a multi-lane link during normal function.</p> <p>See <a href="#">Lane Enable After Powerdown</a> for details on e-enabling powered down lanes.</p> <p>0b - Lane transmitter active 1b - Lane transmitter powered down</p>
13 IF20BIT_EN	<p>20-bit interface enable</p> <p>Default value set by reset configuration.</p> <p>Required value per protocol:</p> <ul style="list-style-type: none"> <li>PCIe 1</li> <li>XFI 1</li> <li>SGMII 0</li> <li>QSGMII 1</li> <li>SATA 1</li> </ul> <p>0b - 10-bit interface 1b - 20-bit interface</p>
14 —	Reserved
15 FIRST_LANE	<p>Indicates this lane is the first (lane 0) of a group of lanes</p> <p>Default/recommended value set by reset configuration</p> <p>0b - Lane a is not lane 0 of the link 1b - Lane a is lane 0 of the link or unused lane</p>
16-19 —	Reserved
20-24 PROTS	<p>Lane Protocol Select. Default value is set by reset configuration</p> <p>All others reserved</p> <p>00000b - PCIe 00001b - SGMII 1000Base-KX QSGMII OC-SGMII 00010b - SATA 01010b - XFI</p>
25-31 —	Reserved

### 33.5.18 General Control Register 1 - Lane a (LNAGCR1 - LNDG CR1)

### 33.5.18.1 Offset

Register	Offset
LNAGCR1	804h
LNBGCR1	844h
LNCGCR1	884h
LNDGCR1	8C4h

### 33.5.18.2 Function

LNmGCR1 contains functional control bits for SerDes lane a.

### 33.5.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RDAT_INV	TDAT_INV	Reserved		OPAD_CTL	Reserved			REIDL_TH				REIDL_EX_SE		REIDL_ET_SE	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	REIDL_EX_MS_B	REIDL_ET_MS_B	REQ_CTL_SN_P	REQ_CDR_SN_P	Reserved				TRSTDIR		REQ_BIN_SN_P	ISLEW_RCT_L		Reserved	OSLEW_RCT_L	
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1

### 33.5.18.4 Fields

Field	Function
0 RDAT_INV	Invert Rx data. Has the same effect as swapping SD_RX[m]_P and SD_RX[m]_N.  Note: this field is read-only when this lane is operating as PCIe.  0b - Rx data is not inverted 1b - Rx data is inverted before it is decoded

Table continues on the next page...

Field	Function
1 TDAT_INV	Invert Tx data. Has the same effect as swapping SD_TX[m]_P and SD_TX[m]_N. 0b - Tx data is not inverted 1b - Tx data is inverted before it is transmitted
2-4 —	Reserved
5 OPAD_CTL	TX Output pad control signal for common mode Note: this field is read-only when this lane is operating as PCIe, SATA, XFI, SGMII or QSGMII. 0b - Transmitter Enabled 1b - Force Transmitter Output to Common Mode
6-8 —	Reserved
9-11 REIDL_TH	Receiver electrical idle detection threshold control Default value set by reset configuration Recommended value per protocol: PCIe: 100 (2.5 Gbaud) SGMII: 001 (1.25 Gbaud) SGMII: 000 (3.125 Gbaud) SATA: 100 (1.5 Gbaud) Others: 000 Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[REIDL_TH_0]. Note: SATA 6 Gbaud setting taken from LNaSSCR1[REIDL_TH_1].
12-13 REIDL_EX_SEL	Exit electrical idle filter select = {REIDL_EX_MSB,REIDL_EX_SEL} Default value set by reset configuration Recommended value per protocol: PCIe: 011 (2.5 Gbaud) SGMII: 011 (1.25 Gbaud) SGMII: 000 (3.125 Gbaud) 1GKX: 000 SATA: 001 (1.5 Gbaud) Others: 000 Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[REIDL_EX_SEL_0]. Note: SATA 6 Gbaud setting taken from LNaSSCR1[REIDL_EX_SEL_1].
14-15 REIDL_ET_SEL	Enter idle filter select = {REIDL_ET_MSB,REIDL_ET_SEL}: Recommended setting per protocol: PCIe: 111 (2.5 Gbaud) SGMII: 100 (1.25 Gbaud) SGMII: 000 (3.125 Gbaud) 1GKX: 000 SATA: 001 (1.5 Gbaud)

Table continues on the next page...

## SerDes register descriptions

Field	Function
	<p>Others: 000</p> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[REIDL_ET_SEL_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[REIDL_ET_SEL_1].</p>
16 REIDL_EX_MSB	<p>Exit idle filter select MSB. See REIDL_EX_SEL for settings.</p> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[REIDL_EX_MSB_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[REIDL_EX_MSB_1].</p>
17 REIDL_ET_MSB	<p>Enter idle filter select MSB. See REIDL_ET_SEL for settings.</p> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[REIDL_ET_MSB_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[REIDL_ET_MSB_1].</p>
18 REQ_CTL_SNP	<p>Initiate snapshot of RX Equalization Control Gaink2, Gaink3, and Offset Registers</p> <p>Recommended Setting per protocol: 0</p>
19 REQ_CDR_SN_P	<p>initiate snapshot of RX Clock/Data Recovery (CDR) Registers</p> <p>Recommended Setting per protocol: 0</p>
20-23 —	Reserved
24 TRSTDIR	<p>Multi-lane protocol Tx clock synchronization control</p> <p>Default value set by reset configuration.</p> <p>Recommended setting per protocol:</p> <p>PCIe: 1</p> <p>Others: 0</p>
25 REQ_BIN_SNP	<p>Initiate snapshot of RX Equalization Control Binning Registers</p> <p>Note: this field is read-only when this lane is operating as PCIe</p>
26-27 ISLEW_RCTL	<p>Slew control for Quadrature Generator</p> <p>Default value set by RCS configuration.</p> <p>Recommended setting per protocol:</p> <p>PCIe: 01 (2.5 Gbaud)</p> <p>SGMII: 01 (1.25 Gbaud)</p> <p>SGMII: 10 (3.125 Gbaud)</p> <p>QSGMII: 01</p> <p>XFI: 01</p> <p>SATA: 10 (1.5 Gbaud)</p> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[ISLEW_RCTL_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[ISLEW_RCTL_1].</p>
28-29 —	Reserved
30-31 OSLEW_RCTL	<p>Phase Interpolator Output clock edge rate control</p> <p>Default value set by reset configuration.</p> <p>Recommended setting per protocol:</p>

Field	Function
	<ul style="list-style-type: none"> <li>PCIe: 01 (2.5 Gbaud)</li> <li>SGMII: 01 (1.25 Gbaud)</li> <li>SGMII: 10 (3.125 Gbaud)</li> <li>QSGMII: 01</li> <li>XFI: 01</li> <li>SATA: 10 (1.5 Gbaud)</li> </ul> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[OSLEW_RCTL_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[OSLEW_RCTL_1].</p>

### 33.5.19 Speed Switch Control Register 0 - Lane a (LNASSCR0 - LNDSSCR0)

#### 33.5.19.1 Offset

Register	Offset
LNASSCR0	80Ch
LNBSSCR0	84Ch
LNCSSCR0	88Ch
LNDSSCR0	8CCh

#### 33.5.19.2 Function

LNmSSCR0 contains control bits for modifying the PCI Express 5 Gbaud and SATA 3 Gbaud behavior of SerDes lane a. It is reserved for other protocols.

### 33.5.19.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	REIDL_TH_0			REIDL_EX_SEL_0		REIDL_ET_SEL_0		REIDL_EX_MSB_0	REIDL_ET_MSB_0	ISLEW_RCTL_0		OSLEW_RCTL_0		RXEQ_BST_0	BASE_WAND_0	
W																
Reset	1	0	0	1	1	0	0	0	0	0	1	0	1	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	OSETOVD6_0		TEQ_TYPE_0		SGN_PREQ_0		SGN_POST1Q_0		RATIO_PST1Q_0					AMP_RED_0		
W																
Reset	0	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0

### 33.5.19.4 Fields

Field	Function
0-2 REIDL_TH_0	Receiver electrical idle detection threshold control  Default settings set per reset configuration.  Recommended settings per protocol:  PCIe: 100 (5 Gbaud)  SATA: 101 (3 Gbaud)  For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_TH].  For SATA 6 Gbaud, see LNaSSCR1[REIDL_TH_1].
3-4 REIDL_EX_SEL_0	Exit electrical idle filter select = {REIDL_EX_MSB_0,REIDL_EX_SEL_0}:  Default settings set per reset configuration.  Recommended settings per protocol:  PCIe: 011 (5 Gbaud)  SATA: 001 (3 Gbaud)  For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_EX_SEL].  For SATA 6 Gbaud, see LNaSSCR1[REIDL_EX_SEL_1].
5-6	Enter idle filter select = {REIDL_ET_MSB_0,REIDL_ET_SEL_0}:  Default settings set per reset configuration.

Table continues on the next page...

Field	Function
REIDL_ET_SEL_0	<p>Recommended settings per protocol:</p> <p>PCIe: 000 (5 Gbaud)</p> <p>SATA: 001 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_ET_SEL].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[REIDL_ET_SEL_1].</p>
7 REIDL_EX_MSB_0	<p>Exit idle filter select MSB. See REIDL_EX_SEL_0 for settings.</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_EX_MSB].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[REIDL_EX_MSB_1].</p>
8 REIDL_ET_MSB_0	<p>Enter idle filter select MSB. See REIDL_ET_SEL_0 for settings.</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_ET_MSB].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[REIDL_ET_MSB_1].</p>
9-10 ISLEW_RCTL_0	<p>Slew control for Quadrature Generator</p> <p>Default settings set per reset configuration.</p> <p>Recommended settings per protocol:</p> <p>PCIe: 01 (5 Gbaud)</p> <p>SATA: 10 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[ISLEW_RCTRL].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[ISLEW_RCTRL_1].</p>
11-12 OSLEW_RCTL_0	<p>Phase Interpolator Output clock edge rate control</p> <p>Default settings set per reset configuration.</p> <p>Recommended settings per protocol:</p> <p>PCIe: 01</p> <p>SATA: 10</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[OSLEW_RCTL].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[OSLEW_RCTL_1].</p>
13 RXEQ_BST_0	<p>Rx Equalization Boost</p> <p>Default value set by reset configuration</p> <p>Recommended value per protocol: 0</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaRECR0[RXEQ_BST].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[RXEQ_BST_1].</p>
14-15 BASE_WAND_0	<p>Baseline Wander Control Select</p> <p>Default settings set per reset configuration.</p> <p>Recommended settings per protocol: 00</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaRECR0[BASE_WAND].</p>

*Table continues on the next page...*

## SerDes register descriptions

Field	Function
	<p>For SATA 6 Gbaud, see LNaSSCR1[BASE_WAND_1].</p> <p>00b - off (8b10b data)      01b - default BinBLW threshold      10b - alternate BinBLW sign      11b - Use RX Eq offset as GainBLW override</p>
16 OSETOVD6_0	<p>Binary Decode of Lane Adaptive Equalization offset initialization or override value.</p> <p>[6] = 1: Double Imposed Offset</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>PCIe: 0 (5 Gbaud)</p> <p>SATA: 0 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[OSETOVD].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[OSETOVD6_1].</p>
17-18 TEQ_TYPE_0	<p>Lane transmit equalization.</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>PCIe: 01 (5 Gbaud)</p> <p>SATA: 01 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[TEQ_TYPE].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[TEQ_TYPE_1].</p> <p>00b - No TX Equalization      01b - 2 Levels of TX Equalization (+1 post cursor)      10b - 3 Levels of TX Equalization (+1 pre-cursor and +1 post-cursor)      11b - Reserved</p>
19 SGN_PREQ_0	<p>Precursor sign</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>PCIe: 0 (5 Gbaud)</p> <p>SATA: 0 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[SGN_PREQ].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[SGN_PREQ_1].</p> <p>0b - Negative Sign (close eye)      1b - Positive Sign (open eye)</p>
20 SGN_POST1Q_0	<p>Post1q sign</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>PCIe: 1 (5 Gbaud)</p> <p>SATA: 1 (3 Gbaud)</p>

Table continues on the next page...

Field	Function
	<p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[SGN_POST1Q].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[SGN_POST1Q_1]</p> <p>0b - Negative Sign (close eye) 1b - Positive Sign (open eye)</p>
21-25 RATIO_PST1Q_0	<p>Ratio of full swing transition bit to first post-cursor.</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol: PCIe: 0_1100 (5 Gbaud) SATA: 0_0010 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[RATIO_PST1Q].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[RATIO_PST1Q_1].</p> <p>Note: this field is read-only when this lane is operating as PCIe</p>
26-31 AMP_RED_0	<p>Overall TX Amplitude Reduction</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol: PCIe: 00_0000 (5 Gbaud) SATA: 00_0111 (3 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[AMP_RED].</p> <p>For SATA 6 Gbaud, see LNaSSCR1[AMP_RED_1].</p> <p>Note: this field is read-only when this lane is operating as PCIe</p>

### 33.5.20 Receive Equalization Control Register 0 - Lane a (LNAR ECR0 - LNDRECR0)

#### 33.5.20.1 Offset

Register	Offset
LNARECR0	810h
LNBRECR0	850h
LNCRECR0	890h
LNDRECR0	8D0h

### 33.5.20.2 Function

LNmRECR0 contains functional control bits for SerDes lane a

### 33.5.20.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved			RXEQ_BS_T	GK2OVD				Reserved				GK3OVD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	GK2OVD_EN	GK3OVD_EN	OSETOVD_EN	Reserved	BASE_WAND		Reserved	Reserved	OSETOVD							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

### 33.5.20.4 Fields

Field	Function
0-2 —	Reserved
3 RXEQ_BST	Rx Equalization Boost Default value set by reset configuration Recommended value per protocol: 0 Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[RXEQ_BST_0]. Note: SATA 6 Gbaud setting taken from LNaSSCR1[RXEQ_BST_1].
4-7 GK2OVD	Binary decode of lane Adaptive Equalization gaink2 initialization or override value. Default value set by reset configuration. Recommended settings per protocol: PCIe: 0000 SGMII: 1111 (1.25 Gbaud) SGMII: 0000 (3.125 Gbaud) QSGMII: 0000 XFI: 0000

Table continues on the next page...

Field	Function
	SATA: 0000
8-11 —	Reserved
12-15 GK3OVD	<p>Binary decode of lane Adaptive Equalization gaink3 initialization or override value.</p> <p>Default value set by reset config.</p> <p>Recommended settings per protocol:</p> <ul style="list-style-type: none"> <li>PCIe: 0000</li> <li>SGMII: 1111 (1.25 Gbaud)</li> <li>SGMII: 0000 (3.125 Gbaud)</li> <li>QSGMII: 0000</li> <li>XFI: 0000</li> <li>SATA: 0000</li> </ul>
16 GK2OVD_EN	<p>Controls source of rx equalization "gaink2" setting.</p> <p>Recommended settings per protocol:</p> <ul style="list-style-type: none"> <li>SGMII: 1 (1.25 Gbaud)</li> <li>SGMII: 0 (3.125 Gbaud)</li> <li>Others: 0 <ul style="list-style-type: none"> <li>0b - Use rxreq adaption derived gaink2</li> <li>1b - Fix gaink2 according to GK2OVD</li> </ul> </li> </ul>
17 GK3OVD_EN	<p>Controls source of rx equalization "gaink3" setting.</p> <p>Recommended settings per protocol:</p> <ul style="list-style-type: none"> <li>SGMII: 1 (1.25 Gbaud)</li> <li>SGMII: 0 (3.125 Gbaud)</li> <li>Others: 0 <ul style="list-style-type: none"> <li>0 Use rxreq adaption derived gaink3</li> <li>1 Fix gaink3 according to GK3OVD</li> </ul> </li> </ul>
18 OSETOVD_EN	<p>Controls source of rx equalization "offset"</p> <p>setting.</p> <p>Default value set by reset config</p> <p>Recommended setting per protocol: 0 <ul style="list-style-type: none"> <li>0b - On release of srds_reset_b, initial rx eq offset is LNaRECR0[OSETOVD] and rx eq loop will begin adjustment at this value</li> <li>1b - rx eq offset is fixed at LNaRECR0[OSETOVD]</li> </ul> </p>
19 —	Reserved
20-21 BASE_WAND	<p>Baseline Wander Control Select</p> <p>Default value set by reset config</p> <p>Recommended setting per protocol:</p> <ul style="list-style-type: none"> <li>XFI: 01</li> <li>Others: 00</li> </ul>

*Table continues on the next page...*

## SerDes register descriptions

Field	Function
	<p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[BASE_WAND_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[BASE_WAND_1].</p> <p>00b - off (8b10b data)      01b - default BinBLW threshold      10b - alternate BinBLW sign      11b - Use OSETOVD[4:0] as GainBLW override</p>
22-23	Reserved
—	
24	Reserved
—	
25-31 OSETOVD	<p>Binary Decode of Lane Adaptive Equalization offset initialization or override value.</p> <p>[6] 1: Double Imposed Offset</p> <p>[5:0] b00 0000: + Maximum Imposed Offset</p> <p>[5:0] b01 1111: No imposed offset</p> <p>[5:0] b11 1111: - Maximum Imposed Offset</p> <p>Note: If BASE_WAND= 11, then use OSETOVD[4:0] as GainBLW Override</p> <p>Default value set by reset config</p> <p>Recommended setting per protocol:</p> <p>XFI: 101_1111</p> <p>Others: 001_1111</p> <p>Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[OSETOVD_0].</p> <p>Note: SATA 6 Gbaud setting taken from LNaSSCR1[OSETOVD_1]</p>

### 33.5.21 Receive Equalization Control Register 1- Lane a (LNAR ECR1 - LNDRECR1)

#### 33.5.21.1 Offset

Register	Offset
LNARECR1	814h
LNBRECR1	854h
LNCRECR1	894h
LNDRECR1	8D4h

### 33.5.21.2 Function

LNmRECR1 contains control and status bits for receiver equalization on lane a.

### 33.5.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved												EQ_BSNP_DN	EQ_CSNP_DN	CDR_SNPDN	
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.21.4 Fields

Field	Function
0-28	Reserved
—	
29 EQ_BSNP_DN	Snapshot of RX EQ Bin Complete
30 EQ_CSNP_DN	Snapshot of RX EQ Ctrl Complete
31 CDR_SNPDN	Snapshot of CDR Loop Complete

## 33.5.22 Transmit Equalization Control Register 0 - Lane a (LNAT ECR0 - LNDTECR0)

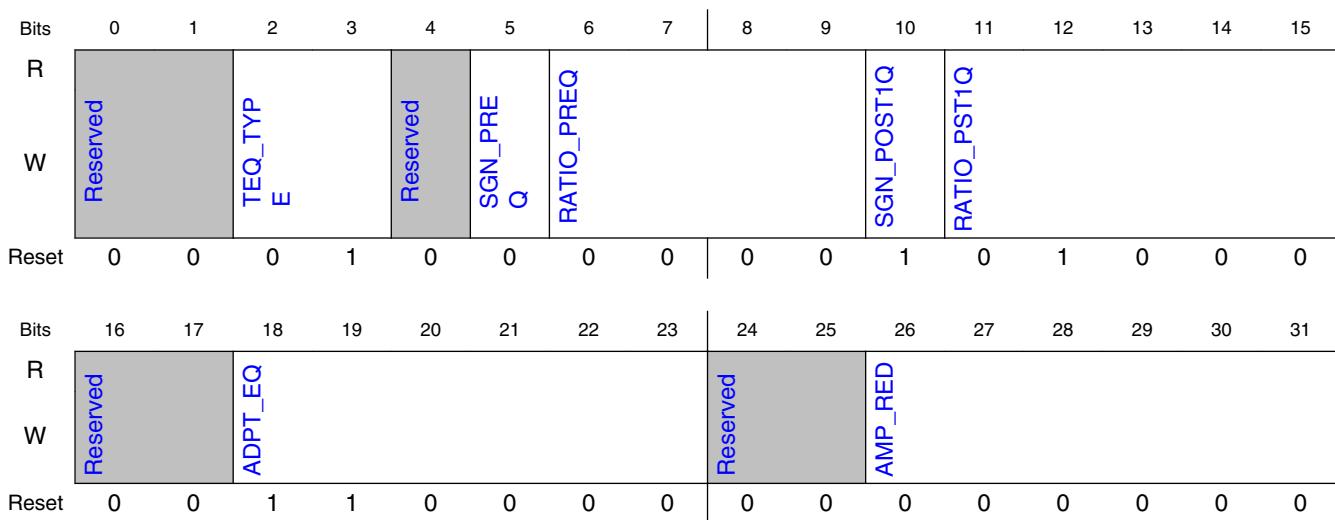
### 33.5.22.1 Offset

Register	Offset
LNATECR0	818h
LNBTECR0	858h
LNCTECR0	898h
LNDTECR0	8D8h

### 33.5.22.2 Function

LNmTECR0 contains Tx equalization control bits for SerDes lane a.

### 33.5.22.3 Diagram



### 33.5.22.4 Fields

Field	Function
0-1 —	Reserved
2-3 TEQ_TYPE	Selects amount/type of Transmit Equalization Default value set by reset configuration. Recommended settings per protocol:

Table continues on the next page...

Field	Function
	<p>SGMII: 00 (1.25 Gbaud)          SGMII: 01 (3.125 Gbaud)          XFI: 01          Others: 01          Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[TEQ_TYPE_0].          Note: SATA 6 Gbaud setting taken from LNaSSCR1[TEQ_TYPE_1]</p> <p>00b - No TX Equalization          01b - 2 Levels of TX Equalization (+1 post cursor)          10b - 3 Levels of TX Equalization (+1 pre-cursor and +1 post-cursor)          11b - Reserved</p>
4	Reserved
—	
5 SGN_PREQ	<p>Precursor sign          Default value set by reset configuration.          Recommended settings per protocol:          XFI: 0          Others: 0          Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[SGN_PREQ_0].          Note: SATA 6 Gbaud setting taken from LNaSSCR1[SGN_PREQ_1]</p> <p>0b - Negative Sign(close eye)          1b - Positive Sign (open eye)</p>
6-9 RATIO_PREQ	<p>Ratio of full swing transition bit to pre-cursor          Default value set by reset configuration.          Recommended settings per protocol:          XFI: 0000          Others: 0000          Note: this field is read-only when this lane is operating as PCIe</p>
10 SGN_POST1Q	<p>Post q Sign          Default value set by reset configuration.          Recommended settings per protocol:          SGMII: 0 (1.25 Gbaud)          SGMII: 1 (3.125 Gbaud)          Others: 1          Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[SGN_POST1Q_0]          Note: SATA 6 Gbaud setting taken from LNaSSCR1[SGN_POST1Q_1].</p> <p>0b - Negative Sign (close eye)          1b - Positive Sign (open eye)</p>
11-15 RATIO_PST1Q	<p>Ratio of full swing transition bit to first post-cursor.          Default value set by reset configuration.          Recommended settings per protocol:</p>

*Table continues on the next page...*

## SerDes register descriptions

Field	Function
	PCIe: 0_1000 (2.5 Gbaud) SGMII: 0_0000 (1.25 Gbaud) SGMII: 0_0110 (3.125 Gbaud) XFI: 0_0011 SATA: 0_0010 (1.5 Gbaud) Others: 0_0110 Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[RATIO_PST1Q_0]. Note: SATA 6 Gbaud setting taken from LNaSSCR1[RATIO_PST1Q_1]. Note: this field is read-only when this lane is operating as PCIe
16-17 —	Reserved
18-23 ADPT_EQ	Transmitter Adjustments for 8G/10G Default value set by reset configuration. Recommended settings per protocol: Others: 11_0000 Note: this field is read-only when this lane is operating as PCIe
24-25 —	Reserved
26-31 AMP_RED	Overall TX Amplitude Reduction Default value set by reset configuration. Recommended settings per protocol: SGMII: 00_0110 (1.25 Gbaud) SGMII: 00_0000 (3.125 Gbaud) 1000BASE-KX: 00_0000 QSGMII: 00_0010 XFI: 00_0111 SATA: 01_0000 (1.5 Gbaud) Others: 00_0000 Note: SATA 3 Gbaud and PCIe 5 Gbaud setting taken from LNaSSCR0[AMP_RED_0]. Note: SATA 6 Gbaud setting taken from LNaSSCR1[AMP_RED_1] Note: this field is read-only when this lane is operating as PCIe

### 33.5.23 Speed Switch Control Register 1- Lane 0 (LNaSSCR1 - LNDSSCR1)

### 33.5.23.1 Offset

Register	Offset
LNASSCR1	81Ch
LNBSSCR1	85Ch
LNCSSCR1	89Ch
LNDSSCR1	8DCh

### 33.5.23.2 Function

LNASSCR1 contains control bits for modifying SATA 6 Gbaud behavior of SerDes lane a. It is reserved for other protocols.

### 33.5.23.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	REIDL_TH_1			REIDL_EX_SEL_1		REIDL_ET_SEL_1		REIDL_EX_MSB_1	REIDL_ET_MSB_1	ISLEW_RCTL_1		OSLEW_RCTL_1		RXEQ_BST_1		BASE_WAND_1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	OSSETOVD6_1	TEQ_TYPE_1		SQN_PREQ_1	SQN_POSTIQ_1	RATIO_PST1Q_1								AMP_RED_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.23.4 Fields

Field	Function
0-2 REIDL_TH_1	Receiver electrical idle detection threshold control Default set by reset configuration.

*Table continues on the next page...*

## SerDes register descriptions

Field	Function
	<p>Recommended settings per protocol:</p> <p>SATA: 000 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_TH].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[REIDL_TH_0].</p>
3-4 REIDL_EX_SEL_1	<p>Exit electrical idle filter select = REIDL_EX_MSB_1  REIDL_EX_SEL_1</p> <p>Default set by reset configuration.</p> <p>Recommended settings per protocol:</p> <p>SATA: 000 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_EX_SEL].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[REIDL_EX_SEL_0].</p>
5-6 REIDL_ET_SEL_1	<p>Enter idle filter select = REIDL_ET_MSB_1  REIDL_ET_SEL_1:</p> <p>Default set by reset configuration.</p> <p>Recommended setting per protocol:</p> <p>SATA: 000 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_ET_SEL].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[REIDL_ET_SEL_0].</p>
7 REIDL_EX_MSB_1	<p>Exit idle filter select MSB. See REIDL_EX_SEL_1 for settings.</p> <p>Default set by reset configuration.</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_EX_MSB].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[REIDL_EX_MSB_0].</p>
8 REIDL_ET_MSB_1	<p>Enter idle filter select MSB. See REIDL_ET_SEL_1 for settings.</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[REIDL_ET_MSB].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[REIDL_ET_MSB_0].</p>
9-10 ISLEW_RCTL_1	<p>Slew control for Quadrature Generator</p> <p>Default value set by reset configuration</p> <p>Recommended Setting per protocol:</p> <p>SATA: 10 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[ISLEW_RCTL].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[ISLEW_RCTL_0].</p>
11-12 OSLEW_RCTL_1	<p>Phase Interpolator Output clock edge rate control</p> <p>Default value set by reset configuration</p> <p>Recommended Setting per protocol:</p> <p>SATA: 10 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaGCR1[OSLEW_RCTL].</p>

*Table continues on the next page...*

Field	Function
13 RXEQ_BST_1	<p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[OSLEW_RCTL_0].</p> <p>Rx Equalization Boost</p> <p>Default value set by reset configuration</p> <p>Recommended value per protocol: 0</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud settings, see LNaRECR0[RXEQ_BST].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud settings, see LNaSSCR0[RXEQ_BST_0].</p>
14-15 BASE_WAND_1	<p>Baseline Wander Control Select</p> <p>Default value set by reset config</p> <p>Recommended setting per protocol:</p> <p>SATA: 00 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud settings, see LNaRECR0[BASE_WAND].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud settings, see LNaSSCR0[BASE_WAND_0].</p> <ul style="list-style-type: none"> <li>00b - off (8b10b data)</li> <li>01b - default BinBLW threshold</li> <li>10b - alternate BinBLW sign</li> <li>11b - Use OSETOVD[4:0] as GainBLW override</li> </ul>
16 OSETOVD6_1	<p>Binary Decode of Lane Adaptive Equalization offset initialization or override value.</p> <p>[6] = 1: Double Imposed Offset</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>SATA: 0 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaRECR0[OSETOVD].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[OSETOVD6_0].</p>
17-18 TEQ_TYPE_1	<p>Lane transmit equalization.</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>SATA: 01 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[TEQ_TYPE].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[TEQ_TYPE_0].</p> <ul style="list-style-type: none"> <li>00b - No TX Equalization</li> <li>01b - 2 Levels of TX Equalization (+1 post cursor)</li> <li>10b - 3 Levels of TX Equalization (+1 pre-cursor and +1 post-cursor)</li> <li>11b - Reserved</li> </ul>
19 SGN_PREQ_1	<p>Precursor sign</p> <p>Default value set by reset configuration</p> <p>Recommended setting per protocol:</p> <p>SATA: 0 (6 Gbaud)</p> <p>For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[SGN_PREQ].</p> <p>For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[SGN_PREQ_0].</p>

*Table continues on the next page...*

## SerDes register descriptions

Field	Function
	0b - Negative Sign (close eye) 1b - Positive Sign (open eye)
20 SGN_POST1Q_1	Post1q sign Default value set by reset configuration Recommended setting per protocol: SATA: 1 (6 Gbaud) For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[SGN_POST1Q]. For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[SGN_POST1Q_0]. 0b - Negative Sign (close eye) 1b - Positive Sign (open eye)
21-25 RATIO_PST1Q_1	Ratio of full swing transition bit to first post-cursor. Default value set by reset configuration Recommended setting per protocol: SATA: 0_0010 (6 Gbaud) For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[AMP_RED]. For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[AMP_RED_0]. Note: this field is read-only when this lane is operating as PCIe
26-31 AMP_RED_1	Overall TX Amplitude Reduction Default value set by reset configuration Recommended setting per protocol: SATA: 00_0000 (6 Gbaud) For PCIe 2.5 Gbaud and SATA 1.5 Gbaud and protocols other than PCIe or SATA, see LNaTECR0[AMP_RED]. For PCIe 5 Gbaud and SATA 3 Gbaud, see LNaSSCR0[AMP_RED_0]. Note: this field is read-only when this lane is operating as PCIe

### 33.5.24 TTL Control Register 0 - Lane a (LNATTLCR0 - LNLTLCR0)

#### 33.5.24.1 Offset

Register	Offset
LNATTLCR0	820h
LNBTTLCR0	860h
LNCTTLCR0	8A0h
LNDTTLCR0	8E0h

### 33.5.24.2 Function

LNmTTLCR0 contains control bits for the Transition Tracking Loop (TTL) on SerDes lane a.

### 33.5.24.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved		FLT_SEL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.5.24.4 Fields

Field	Function
0-1 —	Reserved
2-7 FLT_SEL	Selects the gain 'Kfr', 'Kph' and TTL Edge Counting Window Widths in the CDR Loop for the Lane. Default value set by reset configuration. Recommended values per protocol: SGMII: 11_1001 (1.25 Gbaud) SGMII: 00_0000 (3.125 Gbaud) Others: 00_0000
8-31 —	Reserved

## 33.5.25 Test Control/Status Register 3 - Lane a (LNATCSR3 - LNDTCSR3)

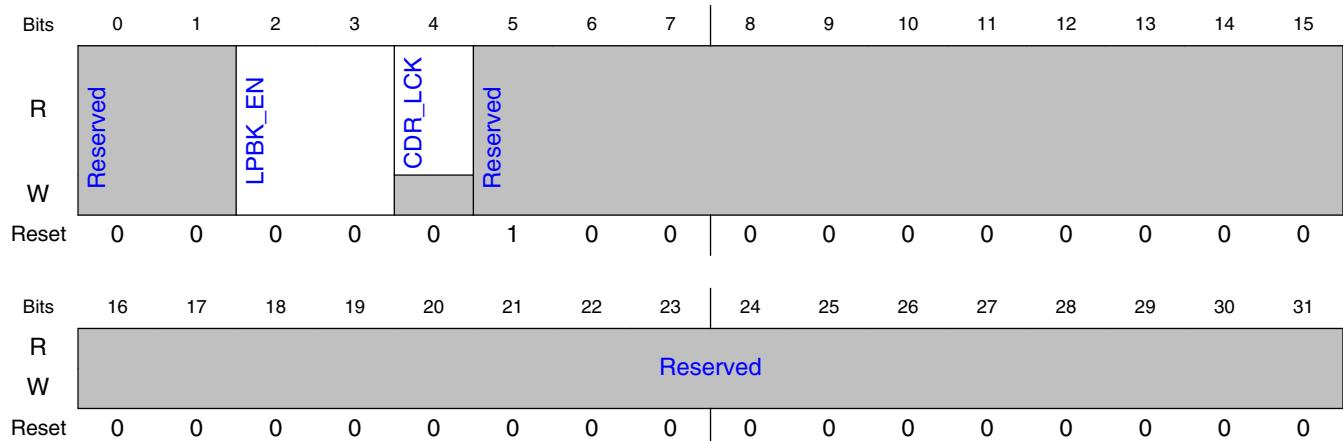
### 33.5.25.1 Offset

Register	Offset
LNATCSR3	83Ch
LNBTCR3	87Ch
LNCTCSR3	8BCh
LNDTCSR3	8FCh

### 33.5.25.2 Function

LNmTCSR3 contains test control and status bits.

### 33.5.25.3 Diagram



### 33.5.25.4 Fields

Field	Function
0-1	Reserved
2-3	Loopback data from TX to RX

Table continues on the next page...

Field	Function
LPBK_EN	All others reserved  Note: Loopback using PCI-Express requires Root Complex configuration. PCI Express End point loopback is not supported.  00b - Application Mode 01b - Loopback Mode
4 CDR_LCK	When asserted, CDR loop has acquired a valid Rx clock
5-31 —	Reserved

## 33.5.26 PEXa Protocol Control Register 0 (PEXACR0 - PEXCCR0)

### 33.5.26.1 Offset

Register	Offset
PEXACR0	1000h
PEXBCR0	1040h
PEXCCR0	1080h

### 33.5.26.2 Function

The PCIe Protocol Control and Status Registers support the control and status bits related to the PCIe PCS layers and PHY control.

PEXnCR0 contains control bits used for the PEXn protocol.

### 33.5.26.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved															
W					RD_SW				Reserved							
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.26.4 Fields

Field	Function
0-3	Reserved
—	
4	Reserved
RD_SW	
5-31	Reserved
—	

## 33.5.27 SGMIIa Protocol Control Register 1 (SGMIIACR1 - SGMIIDCR1)

### 33.5.27.1 Offset

Register	Offset
SGMIIACR1	1804h
SGMIIBCR1	1814h
SGMIICCR1	1824h
SGMIIDCR1	1834h

### 33.5.27.2 Function

SGMIIInCR1 contains control bits used for the SGMIIIn protocol.

### 33.5.27.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	MDEV_PORT					Reserved											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
R	Reserved					SGPCS_EN	Reserved										
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 33.5.27.4 Fields

Field	Function
0-4 MDEV_PORT	<p>MDIO bus port address.</p> <ul style="list-style-type: none"> <li>When accessing using Clause 22, this field is compared to MDIO_CTL[PORT_ADDR] to accept a command.</li> <li>When accessing SGMII using clause 45, the MDIO Ethernet Management Interface register MDIO_CTL[DEV_ADDR], must be 03h and this field is compared to MDIO_CTL[PORT_ADDR] to accept a command. The register address (starting at 8000h) is specified in MDIO_ADDR.</li> </ul> <p>Default value: 0</p> <p>Note: software must wait at least 3 platform clocks after changing this value before performing any MDIO accesses to the SGMIIa PCS.</p>
5-19 —	Reserved
20 SGPCS_EN	<p>SGMII PCS enable</p> <p>Recommended value: 1 if SGMIIa is enabled, else 0</p> <p>Default value is set by reset</p> <p>Note: must be set to 1 as part of reconfiguring a port from XFI to SGMII or 1000Base-KX operation</p>
21-31 —	Reserved

## 33.5.28 SGMIIa Protocol Control Register 3 (SGMIIACR3 - SGMIIDCR3)

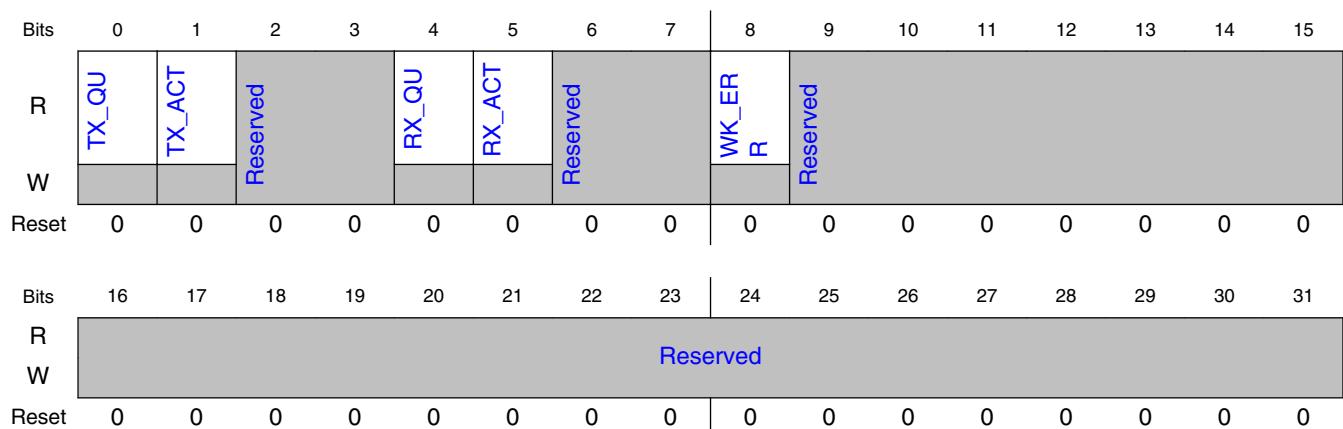
### 33.5.28.1 Offset

Register	Offset
SGMIIACR3	180Ch
SGMIIBCR3	181Ch
SGMIICCR3	182Ch
SGMIIIDCR3	183Ch

### 33.5.28.2 Function

SGMIIInCR3 contains control bits used for the SGMIIIn protocol.

### 33.5.28.3 Diagram



### 33.5.28.4 Fields

Field	Function
0 TX_QU	Tx quiet LPI state (read-only). Tx is in electrical idle.

Table continues on the next page...

Field	Function
1 TX_ACT	Tx active LPI state (read-only)
2-3 —	reserved
4 RX_QU	Rx quiet LPI state (read-only). Rx is in electrical idle
5 RX_ACT	Rx active LPI state (read-only)
6-7 —	reserved
8 WK_ERR	Error on LPI wake (read-only)
9-31 —	Reserved

### 33.5.29 QSGMIIa Protocol Control Register 1 (QSGMIIACR1 - QSGMIIIBCR1)

#### 33.5.29.1 Offset

Register	Offset
QSGMIIACR1	1884h
QSGMIIIBCR1	1894h

#### 33.5.29.2 Function

QSGMIIInCR1 contains control bits used for the QSGMIIIn protocol.

### 33.5.29.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MDEV_PORT			Reserved												
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.29.4 Fields

Field	Function
0-2 MDEV_PORT	Device port ID. Compared to upper 3 bits of MDIO_CTL[PORT_ADDR] to accept a command. The lower 2 bits of the PHY address map the SGMII port within the QSGMII PCS.  Default value: 0  Note: software must wait at least 3 platform clocks after changing this value before performing any MDIO accesses to the QSGMIIa PCS.
3-31 —	Reserved

## 33.5.30 QSGMIIa Protocol Control Register 3 (QSGMIIACR3 - QSGMIIBCR3)

### 33.5.30.1 Offset

Register	Offset
QSGMIIACR3	188Ch
QSGMIIBCR3	189Ch

### 33.5.30.2 Function

QSGMIIInCR3 contains control/status bits for the QSGMIIIn protocol.

### 33.5.30.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved												WK_ERR0	WK_ERR1	WK_ERR2	WK_ERR3
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RX_QU0	RX_QU1	RX_QU2	RX_QU3	RX_ACT0	RX_ACT1	RX_ACT2	RX_ACT3	TX_QU0	TX_QU1	TX_QU2	TX_QU3	TX_ACT0	TX_ACT1	TX_ACT2	TX_ACT3
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.30.4 Fields

Field	Function
0-11	Reserved
—	
12 WK_ERR0	Port 0 Error on LPI wake (read-only)
13 WK_ERR1	Port 1 Error on LPI wake (read-only)
14 WK_ERR2	Port 2 Error on LPI wake (read-only)
15 WK_ERR3	Port 3 Error on LPI wake (read-only)
16 RX_QU0	Port 0 Rx quiet LPI state (read-only)
17 RX_QU1	Port 1 Rx quiet LPI state (read-only)
18 RX_QU2	Port 2 Rx quiet LPI state (read-only)
19 RX_QU3	Port 3 Rx quiet LPI state (read-only)
20	Port 0 Rx active LPI state (read-only)

Table continues on the next page...

## SerDes register descriptions

Field	Function
RX_ACT0	
21	Port 1 Rx active LPI state (read-only)
RX_ACT1	
22	Port 2 Rx active LPI state (read-only)
RX_ACT2	
23	Port 3 Rx active LPI state (read-only)
RX_ACT3	
24	Port 0 Tx quiet LPI state (read-only)
TX_QU0	
25	Port 1 Tx quiet LPI state (read-only)
TX_QU1	
26	Port 2 Tx quiet LPI state (read-only)
TX_QU2	
27	Port 3 Tx quiet LPI state (read-only)
TX_QU3	
28	Port 0 Tx active LPI state (read-only)
TX_ACT0	
29	Port 1 Tx active LPI state (read-only)
TX_ACT1	
30	Port 2 Tx active LPI state (read-only)
TX_ACT2	
31	Port 3 Tx active LPI state (read-only)
TX_ACT3	

### 33.5.31 XFIa Protocol Control Register 1 (XFIACR1 - XFIBCR1)

#### 33.5.31.1 Offset

Register	Offset
XFIACR1	1984h
1994h	

#### 33.5.31.2 Function

XFIInCR1 contains control bits used for the XFIIn protocol

### 33.5.31.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	MDEV_PORT				Reserved											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.31.4 Fields

Field	Function
0-4 MDEV_PORT	MDIO bus port address. Compared to the MDIO_CTL[PORT_ADDR] to accept a command. Default value: 0  Note: software must wait at least 3 platform clocks after changing this value before performing any MDIO accesses to the SGMIIn PCS.
5-31 —	Reserved

## 33.5.32 XFIa Protocol Control Register 3 (XFIACR3 - XFIBCR3)

### 33.5.32.1 Offset

Register	Offset
XFIACR3	198Ch
199Ch	

### 33.5.32.2 Function

XFIInCR3 contains control bits used for the XFIIn rotocol.

### 33.5.32.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TX_QU	TX_ACT	Reserved		RX_QU	RX_ACT	Reserved		WK_ER		TX_ALRT		Reserved			
W									R							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.5.32.4 Fields

Field	Function
0 TX_QU	Tx quiet LPI state. Read-only
1 TX_ACT	Tx active LPI state. Read-only
2-3 —	Reserved
4 RX_QU	Rx quiet LPI state. Read-only
5 RX_ACT	Rx active LPI state. Read-only
6-7 —	Reserved
8 WK_ERR	LPI wake error. Read-only
9 TX_ALRT	LPI Tx alert. Read-only
10-31 —	Reserved

## 33.6 MDIO register spaces

The SerDes module also contains MDIO slave ports containing registers for each PCS module.

The PCS MDIO registers are accessed through an indirect method from the MDIO Ethernet management interface registers in the Ethernet MAC controllers. There are two methods used to access the PCS MDIO registers—dubbed Clause 45 and Clause 22 (from the IEEE 802.3 sub clauses that define them), depending on the protocol.

The Clause 45 protocols are subdivided into separate spaces selected by the device address. The following table shows the Clause 45 register spaces:

**Table 33-7. Clause 45 register spaces**

XFI MDIO registers			
Register space	Device address	Notes	See
XFI PMA/PMD registers	01h		<a href="#">MDIO_XFI_PMD register descriptions</a>
XFI PCS registers	03h		<a href="#">MDIO_XFI_PCS register descriptions</a>
XFI Auto-Negotiation registers	07h		<a href="#">MDIO_XFI_AN register descriptions</a>
XFI Vendor-Specific 1 registers	1Eh		<a href="#">MDIO_XFI_VENDOR_SPEC register descriptions</a>

**Table 33-7. Clause 45 register spaces**

1000Base-KX MDIO registers			
Register space	Device address	Notes	See
1000Base-KX PCS registers	03h	Includes vendor-specific alias of Clause 22 SGMII registers starting at register address 8000h	<a href="#">MDIO_KX_PCS register descriptions</a>
1000Base-KX Auto-Negotiation registers	07h		<a href="#">MDIO_KX_AN register descriptions</a>
1000Base-KX Vendor-Specific 1 registers	1Dh		<a href="#">MDIO_KX_VENDOR_SPEC register descriptions</a>

The following table shows the Clause 22 register spaces:

**Table 33-8. Clause 22 registers**

Registers	Notes	See
SGMII		<a href="#">MDIO_SGMII register descriptions</a>

*Table continues on the next page...*

**Table 33-8. Clause 22 registers (continued)**

Registers	Notes	See
QSGMII	For QSGMII, the most-significant 3 bits of MDIO_CTL[PHY_ADDR] are compared to QSGMIIInCR1[MDEV_PORT]. The least-significant 2 bits are used to select the specific QSGMII PCS port (0, 1, 2, or 3).	<a href="#">MDIO_QSGMII register descriptions</a>

**NOTE**

Throughout the MDIO register sections:

- Each MDIO register is 16 bits.
- Register offsets are 16-bit offsets, not byte offsets.
- Addresses not listed are reserved.
- Read accesses to reserved addresses return 0x0000.

### 33.6.1 MDIO\_XFI\_PMD register descriptions

The XFI PMA/PMD register space is selected when the associated XFInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 01h.

#### 33.6.1.1 MDIO\_XFI\_PMD memory map

MDIO\_XFI\_PMD base address: 0h

Offset	Register	Width (in bits)	Access	Reset value
0h	XFI PMD Control 1 (XFI_PMD_CR1)	16	RW	0000h
AAh	XFI 10GBASE-R FEC Ability (XFI_10GR_FEC_ABIL)	16	RO	0001h
ABh	XFI 10GBASE-R FEC Control (XFI_10GR_FEC_CTL)	16	RW	0000h
ACh	XFI 10GBASE-R FEC Corrected Blocks Counter Lower (XFI_10GR_FEC_COR_BLK_CNT_L)	16	RO	0000h
ADh	XFI 10GBASE-R FEC Corrected Blocks Counter Upper (XFI_10GR_FEC_COR_BLK_CNT_U)	16	ROZ	0000h
AEh	XFI 10GBASE-R FEC Uncorrected Blocks Counter Lower (XFI_10GR_FEC_UNCOR_BLK_CNT_L)	16	RO	0000h
AFh	XFI 10GBASE-R FEC Uncorrected Blocks Counter Upper (XFI_10GR_FEC_UNCOR_BLK_CNT_U)	16	ROZ	0000h
8000h	XFI 10GBASE-R Vendor-specific PMA Status (XFI_VEND_SPEC_PMA_STAT)	16	RO	0000h

### 33.6.1.2 XFI PMD Control 1 (XFI\_PMD\_CR1)

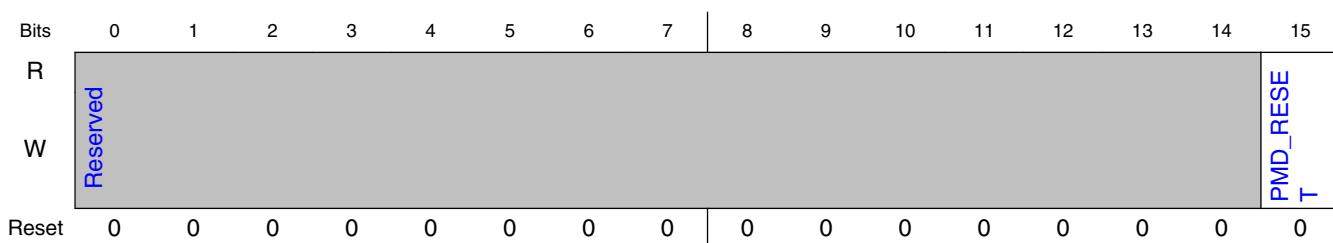
#### 33.6.1.2.1 Offset

Register	Offset
XFI_PMD_CR1	0h

#### 33.6.1.2.2 Function

XFI PMD Control 1 register contains global controls for the XFI PMD module.

#### 33.6.1.2.3 Diagram



#### 33.6.1.2.4 Fields

Field	Function
0-14	Reserved
—	
15 PMD_RESET	PMD Reset

### 33.6.1.3 XFI 10GBASE-R FEC Ability (XFI\_10GR\_FEC\_ABIL)

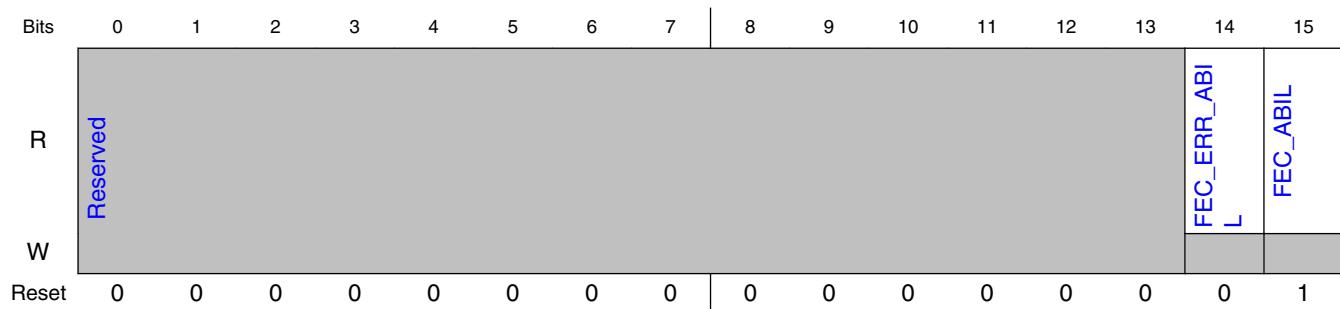
#### 33.6.1.3.1 Offset

Register	Offset
XFI_10GR_FEC_ABIL	AAh

### 33.6.1.3.2 Function

XFI 10GBASE-R FEC Ability Register contains the status bits showing the PCS FEC capability.

### 33.6.1.3.3 Diagram



### 33.6.1.3.4 Fields

Field	Function
0-13 —	Reserved
14 FEC_ERR_ABIL	10GBASE-R FEC Error Indication Ability 10GBASE-R FEC Error Indication Ability Set to 0 to indicate that the FEC is not able to report FEC decoding errors to the PCS layer
15 FEC_ABIL	10GBase-R FEC Ability 10GBase-R FEC Ability Set to 1 to indicate this PCS implements the FEC functions

## 33.6.1.4 XFI 10GBASE-R FEC Control (XFI\_10GR\_FEC\_CTL)

### 33.6.1.4.1 Offset

Register	Offset
XFI_10GR_FEC_CTL	ABh

### 33.6.1.4.2 Function

XFI 10GBASE-R FEC Control Register contains the control bits for the PCS FEC capability.

### 33.6.1.4.3 Diagram



### 33.6.1.4.4 Fields

Field	Function
0-13	Reserved
—	
14 FEC_ENAB_ER R	10GBASE-R FEC Enable Error Indication 10GBASE-R FEC Error Indication Ability Set to 0 to indicate that the FEC is not able to report FEC decoding errors to the PCS layer
15 FEC_ENAB	10GBase-R FEC Enable 10GBase-R FEC Enable 0: FEC is not enabled 1: FEC is enabled

### 33.6.1.5 XFI 10GBASE-R FEC Corrected Blocks Counter Lower (XFI\_10GR\_FEC\_COR\_BLK\_CNT\_L)

#### 33.6.1.5.1 Offset

Register	Offset
XFI_10GR_FEC_COR_BLK_CNT_L	ACh

### 33.6.1.5.2 Function

XFI 10GBASE-R FEC Corrected Blocks Counter Lower contains bits 15:0 of the FEC corrected blocks counter.

### 33.6.1.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									COR_BL_CNT_L							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.1.5.4 Fields

Field	Function
0-15	10GBase-R FEC Corrected Bk=locks Counter Lower
COR_BL_CNT_L	Number of corrected FEC blocks, lower 16 bits
	Reset to 0 when a read is performed on the upper bits. The counter does not roll over to 0 when the value 0xFFFF_FFFF is reached

## 33.6.1.6 XFI 10GBASE-R FEC Corrected Blocks Counter Upper (XFI\_10GR\_FEC\_COR\_BLK\_CNT\_U)

### 33.6.1.6.1 Offset

Register	Offset
XFI_10GR_FEC_COR_BLK_CNT_U	ADh

### 33.6.1.6.2 Function

XFI 10GBASE-R FEC Corrected Blocks Counter Upper contains bits 31:16 of the FEC corrected blocks counter.

### 33.6.1.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.1.6.4 Fields

Field	Function
0-15	10GBase-R FEC Corrected Bk=locks Counter Upper
COR_BL_CNT_U	Number of corrected FEC blocks, upper 16 bits
	Reset to 0 when a read is performed. The counter does not roll over to 0 when the value 0xFFFF_FFFF is reached

## 33.6.1.7 XFI 10GBASE-R FEC Uncorrected Blocks Counter Lower (XFI\_10GR\_FEC\_UNCOR\_BLK\_CNT\_L)

### 33.6.1.7.1 Offset

Register	Offset
XFI_10GR_FEC_UNCOR_BLK_CNT_L	AЕh

### 33.6.1.7.2 Function

XFI 10GBASE-R FEC Uncorrected Blocks Counter Lower contains bits 15:0 of the FEC uncorrected blocks counter.

### 33.6.1.7.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									UNCOR_BLK_CNT_L							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.1.7.4 Fields

Field	Function
0-15	10GBase-R FEC Uncorrected Bk=locks Counter Lower
UNCOR_BL_CNT_L	Number of uncorrected FEC blocks, lower 16 bits
T_L	Reset to 0 when a read is performed on the upper bits. The counter does not roll over to 0 when the value 0xFFFF_FFFF is reached

### 33.6.1.8 XFI 10GBASE-R FEC Uncorrected Blocks Counter Upper (XFI\_10GR\_FEC\_UNCOR\_BLK\_CNT\_U)

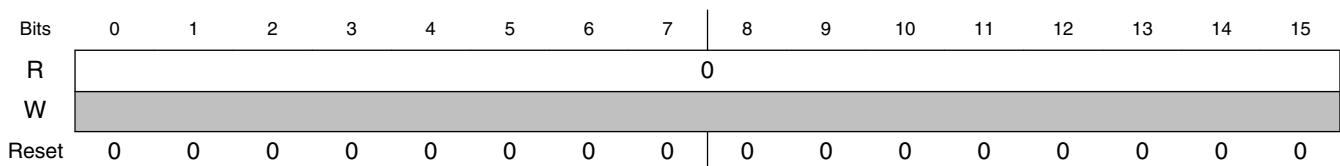
#### 33.6.1.8.1 Offset

Register	Offset
XFI_10GR_FEC_UNCOR_BLK_CNT_U	AFh

#### 33.6.1.8.2 Function

XFI 10GBASE-R FEC Uncorrected Blocks Counter Upper contains bits 31:16 of the FEC uncorrected blocks counter.

#### 33.6.1.8.3 Diagram



#### 33.6.1.8.4 Fields

Field	Function
0-15	10GBase-R FEC Uncorrected Bk=locks Counter Upper
UNCOR_BL_CNT_U	Number of uncorrected FEC blocks, upper 16 bits
T_U	Reset to 0 when a read is performed. The counter does not roll over to 0 when the value 0xFFFF_FFFF is reached

### 33.6.1.9 XFI 10GBASE-R Vendor-specific PMA Status (XFI\_VEND\_SPEC\_PMA\_STAT)

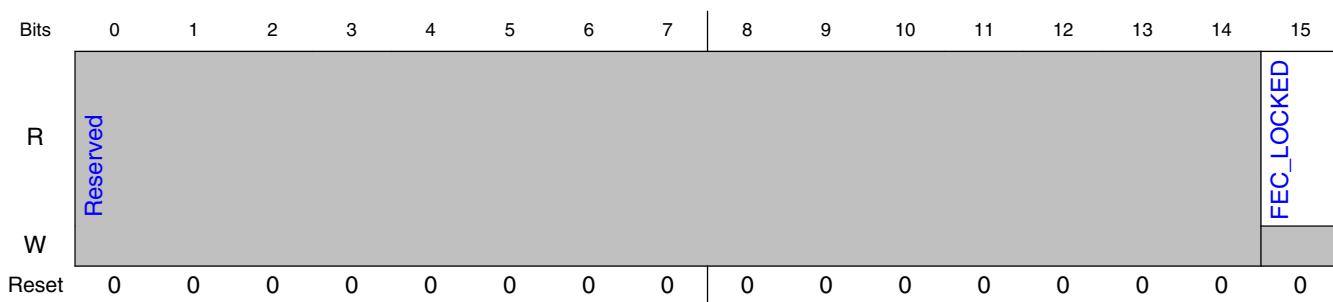
#### 33.6.1.9.1 Offset

Register	Offset
XFI_VEND_SPEC_PMA_STAT	8000h

#### 33.6.1.9.2 Function

XFI 10GBASE-R Vendor-specific PMA Status.

#### 33.6.1.9.3 Diagram



#### 33.6.1.9.4 Fields

Field	Function
0-14	Reserved
—	
15 FEC_LOCKED	10GBase-R FEC Locked FEC Receiver Locked 0: FEC receiver is not locked 2: FEC receiver is locked

## 33.6.2 MDIO\_XFI\_PCS register descriptions

The XFI PCS register space is selected when the associated XFIInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 03h.

### 33.6.2.1 MDIO\_XFI\_PCS memory map

MDIO\_XFI\_PCS base address: 0h

Offset	Register	Width (in bits)	Access	Reset value
0h	XFI PCS Control 1 (XFI_PCS_CR1)	16	RW	2000h
1h	XFI PCS Status 1 (XFI_PCS_SR1)	16	RO	0002h
2h	XFI PCS Device Identifier Upper (XFI_PCS_DEV_ID_H)	16	RO	0083h
3h	XFI PCS Device Identifier Lower (XFI_PCS_DEV_ID_L)	16	RO	E400h
4h	XFI PCS Speed Ability (XFI_PCS_SPEED_ABIL)	16	RO	0001h
5h	XFI PCS Devices In Package 0 (XFI_PCS_DEV_PRES0)	16	RO	008Ah
6h	XFI PCS Devices in Package 1 (XFI_PCS_DEV_PRES1)	16	RO	0000h
7h	XFI 10G PCS Control 2 (XFI_PCS_CR2)	16	RO	000Bh
8h	XFI 10G PCS Status 2 (XFI_PCS_SR2)	16	RO	8001h
Eh	XFI PCS Package Identifier Upper (XFI_PCS_PKG_ID_H)	16	RO	0083h
Fh	XFI PCS Package Identifier Lower (XFI_PCS_PKG_ID_L)	16	RO	E400h
20h	XFI 10GBASE-R PCS Status 1 (XFI_PCS_10GR_SR1)	16	RO	0000h
21h	XFI 10GBASE-R PCS Status 2 (XFI_PCS_10GR_SR2)	16	RO	0000h
22h	XFI 10GBASE-R PCS Test Pattern Seed A 0 (XFI_PCS_TP_SEED_A0)	16	RW	0000h
23h	XFI 10GBASE-R PCS Test Pattern Seed A 1 (XFI_PCS_TP_SEED_A1)	16	RW	0000h
24h	XFI 10GBASE-R PCS Test Pattern Seed A 2 (XFI_PCS_TP_SEED_A2)	16	RW	0000h
25h	XFI 10GBASE-R PCS Test Pattern Seed A 3 (XFI_PCS_TP_SEED_A3)	16	RW	0000h
26h	XFI 10GBASE-R PCS Test Pattern Seed B 0 (XFI_PCS_TP_SEED_B0)	16	RW	0000h
27h	XFI 10GBASE-R PCS Test Pattern Seed B 1 (XFI_PCS_TP_SEED_B1)	16	RW	0000h
28h	XFI 10GBASE-R PCS Test Pattern Seed B 2 (XFI_PCS_TP_SEED_B2)	16	RW	0000h
29h	XFI 10GBASE-R PCS Test Pattern Seed B 3 (XFI_PCS_TP_SEED_B3)	16	RW	0000h
2Ah	XFI 10GBASE-R PCS Test Pattern Control (XFI_PCS_TP_CR)	16	RW	0000h
2Bh	XFI 10GBASE-R PCS Test Pattern Error Counter (XFI_PCS_TP_E_RR_CNT)	16	RO	0000h
8000h	Vendor Specific PCS Status (XFI_PCS_VENDOR_SR)	16	RO	0000h

## 33.6.2.2 XFI PCS Control 1 (XFI\_PCS\_CR1)

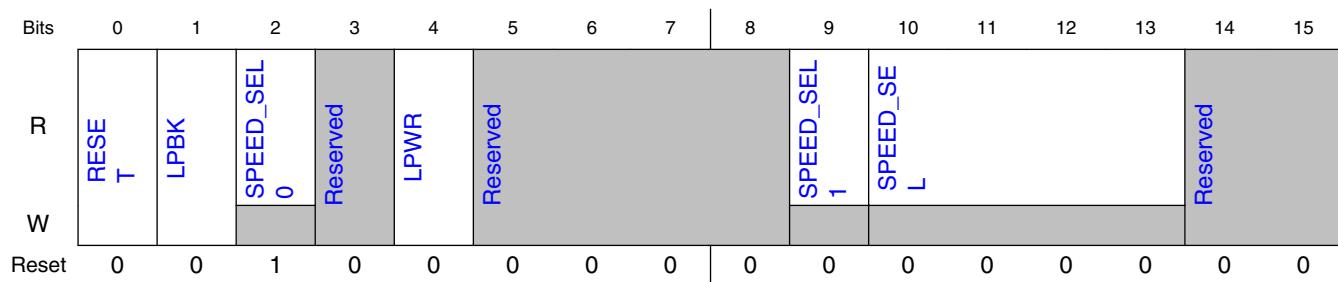
### 33.6.2.2.1 Offset

Register	Offset
XFI_PCS_CR1	0h

### 33.6.2.2.2 Function

The PCS Control 1 Register contains controls for the XFI PCS

### 33.6.2.2.3 Diagram



### 33.6.2.2.4 Fields

Field	Function
0 RESET	Reset Reset Self Clearing bit. 0b - PCS is not in reset 1b - PCS transmit and receive functions are being reset
1 LPBK	Loopback Loopback Enable 0b - Normal function 1b - XGMII Tx data is returned to XGMII Rx
2 SPEED_SEL0	Speed Selection Speed Selection Read only

Table continues on the next page...

## MDIO register spaces

Field	Function
13:6	
3 —	Reserved
4 LPWR	Low Power Operation Low Power Operation 0b - Normal function 1b - Low power operation: PCS is in reset state and most functions are disabled
5-8 —	Reserved
9 SPEED_SEL1	Speed Selection 1 See SPEED_SEL1 Read only
10-13 SPEED_SEL	Speed Selection Speed Selection Read only 0000b - 10 Gbps
14-15 —	Reserved

### 33.6.2.3 XFI PCS Status 1 (XFI\_PCS\_SR1)

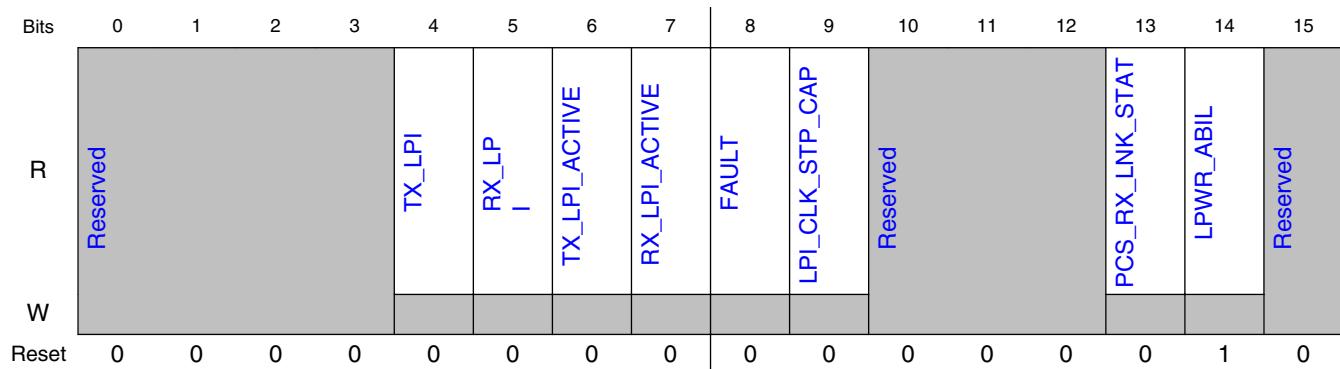
#### 33.6.2.3.1 Offset

Register	Offset
XFI_PCS_SR1	1h

#### 33.6.2.3.2 Function

The XFI PCS Status 1 Register contains status bits for the XFI PCS

### 33.6.2.3.3 Diagram



### 33.6.2.3.4 Fields

Field	Function
0-3 —	Reserved
4 TX_LPI	<p>Tx LPI</p> <p>Once set, stays set until register read</p> <p>Note: for this device Tx LPI state does not Transmit LPI</p> <p>0b - normal operation. 1b - transmit is or was in LPI (EEE)</p>
5 RX_LPI	<p>Rx LPI</p> <p>Receive LPI</p> <p>Once set, stays set until register read</p> <p>0b - normal operation. 1b - receive is or was in LPI state (EEE)</p>
6 TX_LPI_ACTIVE	<p>TX LPI Active</p> <p>Transmit LPI Active</p> <p>0b - normal operation 1b - transmit is currently in LPI state (EEE)</p>
7 RX_LPI_ACTIVE	<p>RX LPI Active</p> <p>Receive LPI Active</p> <p>0b - normal operation 1b - receive is currently in LPI state (EEE).</p>
8 FAULT	<p>Fault</p> <p>Fault</p> <p>0b - no fault condition detected 1b - fault condition detected</p>
9	LPI Clock Stop Capability

Table continues on the next page...

## MDIO register spaces

Field	Function
LPI_CLK_STP_CAP	Clock Stop Capable: 0b - MAC is not capable of stopping the clock during LPI
10-12 —	Reserved
13 PCS_RX_LNK_STAT	PCS Rx Link Status PCS Receive Link Status Once cleared, stays cleared until register read 0b - indicates that the PCS receive link is or was down 1b - indicates that the PCS receive link is up.
14 LPWR_ABIL	Low Power Ability Low Power Ability 1b - The PCS implements a low power mode, meaning bit setting 11 of PCS Control register 1 register is supported and allows to place the PCS in a reset state.
15 —	Reserved

### 33.6.2.4 XFI PCS Device Identifier Upper (XFI\_PCS\_DEV\_ID\_H)

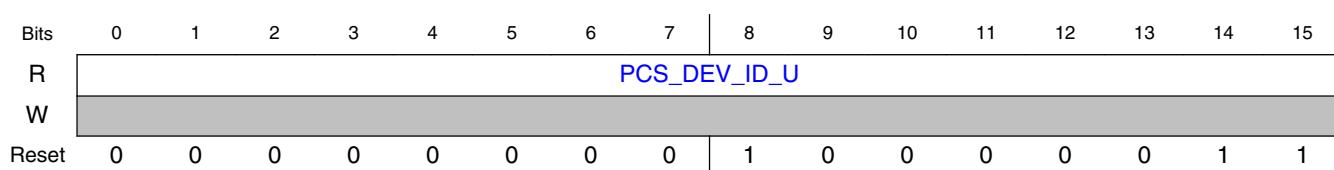
#### 33.6.2.4.1 Offset

Register	Offset
XFI_PCS_DEV_ID_H	2h

#### 33.6.2.4.2 Function

The XFI PCS Device Identifier Upper Register contains the upper half of the PCS Device Identifier

#### 33.6.2.4.3 Diagram



### 33.6.2.4.4 Fields

Field	Function
0-15	PCS Device ID Upper
PCS_DEV_ID_U	PCS Device ID Upper: 15:0 - OIU[3:18]

### 33.6.2.5 XFI PCS Device Identifier Lower (XFI\_PCS\_DEV\_ID\_L)

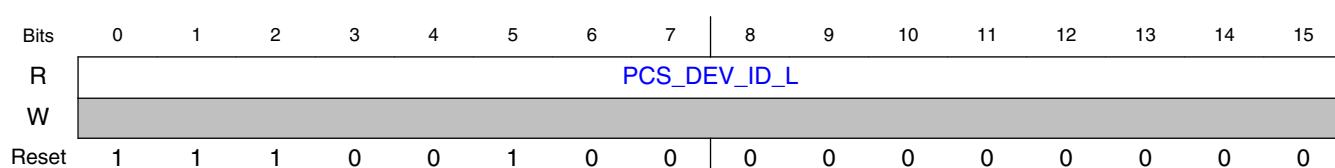
#### 33.6.2.5.1 Offset

Register	Offset
XFI_PCS_DEV_ID_L	3h

#### 33.6.2.5.2 Function

The XFI PCS Device Identifier Lower Register contains the lower half of the PCS Device Identifier

#### 33.6.2.5.3 Diagram



#### 33.6.2.5.4 Fields

Field	Function
0-15	PCS Device ID Lower
PCS_DEV_ID_L	PCS Device ID Lower: 15:10 - OIU[19:24] 9:4 - Manufacturer's Model Number 3:0 - Revision Number

### 33.6.2.6 XFI PCS Speed Ability (XFI\_PCS\_SPEED\_ABIL)

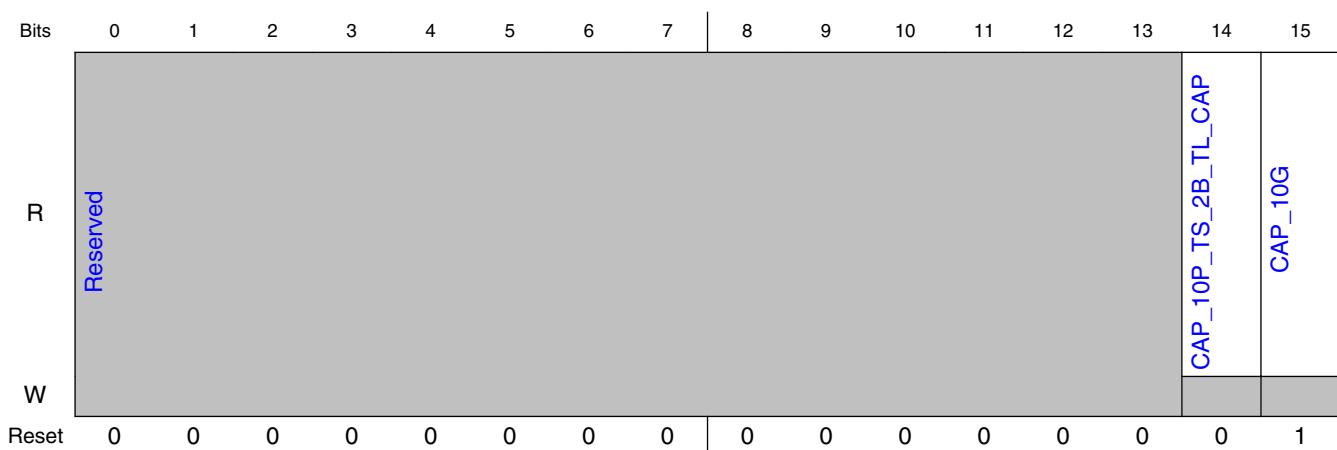
#### 33.6.2.6.1 Offset

Register	Offset
XFI_PCS_SPEED_ABIL	4h

#### 33.6.2.6.2 Function

The XFI PCS Speed Ability Register contains ability bits for the 10GBASE-R PCS

#### 33.6.2.6.3 Diagram



#### 33.6.2.6.4 Fields

Field	Function
0-13 —	Reserved
14 CAP_10P_TS_2B_TL_CAP	10Pass-TS/2Base-TL Capable 10PASS-TS/2BASE-TL Capable 0b - The PCS is not capable of operating as the 10P/2B PCS
15 CAP_10G	1-G Capable 10G Capable 1b - PCS is capable of operating at 10Gbps.

### 33.6.2.7 XFI PCS Devices In Package 0 (XFI\_PCS\_DEV\_PRES0)

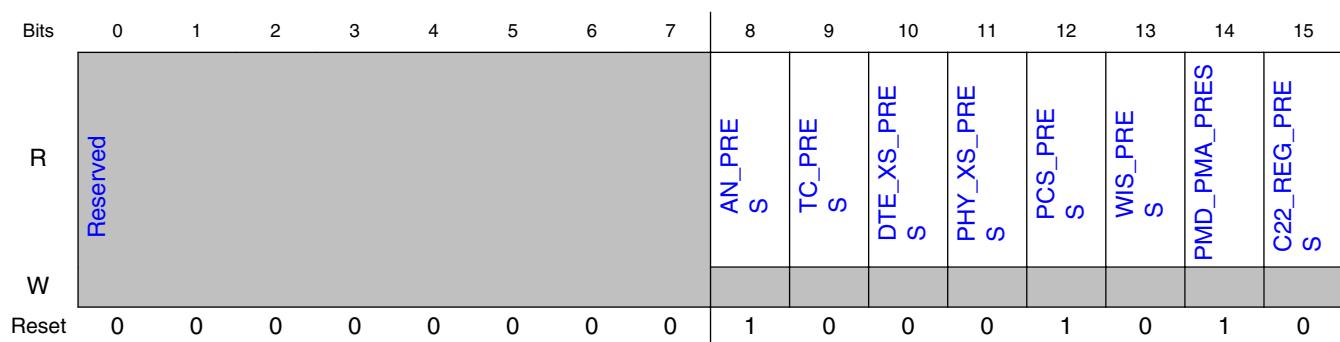
#### 33.6.2.7.1 Offset

Register	Offset
XFI_PCS_DEV_PRES0	5h

#### 33.6.2.7.2 Function

The XFI PCS Devices in Package 0 Register contains half of the devices present status

#### 33.6.2.7.3 Diagram



#### 33.6.2.7.4 Fields

Field	Function
0-7 —	Reserved
8 AN_PRES	AN Present Auto-negotiation Present 1b - the PCS implements the Auto-Negotiation function
9 TC_PRES	TC Present TC Present 0b - the PCS does not implement TC functions
10 DTE_XS_PRES	DTE XS Present DTE XS Present 0b - the PCS does not implement DTE XS functions

Table continues on the next page...

## MDIO register spaces

Field	Function
11 PHY_XS_PRES	PHY XS Present PHY XS Present 0b - the PCS does not implement PHY XS functions.
12 PCS_PRES	PSC Present PCS Present 1b - the PCS implements PCS functions
13 WIS_PRES	WIS Present WIS Present 0b - the PCS does not implement a WIS function.
14 PMD_PMA_PR ES	PMD/PMA Present PMD/PMA Present 1b - the PCS implements PMD/PMA functions (Link Training and FEC functions)
15 C22_REG_PRE S	Clause 22 Registers Present Clause 22 Registers Present 0b - the PCS does not implement the Clause 22 Registers.

### 33.6.2.8 XFI PCS Devices in Package 1 (XFI\_PCS\_DEV\_PRES1)

#### 33.6.2.8.1 Offset

Register	Offset
XFI_PCS_DEV_PRES1	6h

#### 33.6.2.8.2 Function

The XFI PCS Devices in Package 1 Register contains half of the devices present status

### 33.6.2.8.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	VEND_SPEC_DEV2_PRE_S	VEND_SPEC_DEV1_PRE_S	CLAUSE22_EXT_PRE_S	Reserved												
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.2.8.4 Fields

Field	Function
0 VEND_SPEC_DEV2_PRESENT	Vendor Specific Device 2 Present Vendor Specific Device 2 Present 0b - the PCS does not implement any vendor specific device.
1 VEND_SPEC_DEV1_PRESENT	Vendor Specific Device 1 Present Vendor Specific Device 1 Present 0b - the PCS does not implement any vendor specific device.
2 CLAUSE22_EXTENSION_PRESENT	Clause 22 Extension Present Clause 22 Extension Present 0b - the PCS does not implement Clause 22 extensions.
3-15 —	Reserved

## 33.6.2.9 XFI 10G PCS Control 2 (XFI\_PCS\_CR2)

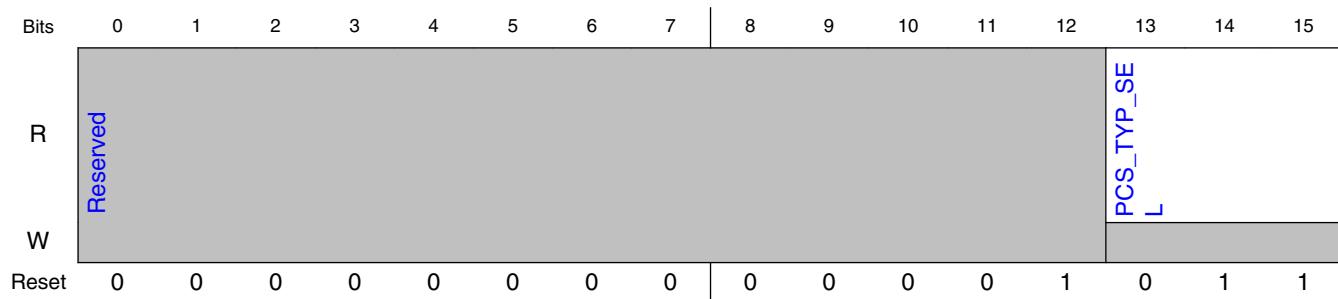
### 33.6.2.9.1 Offset

Register	Offset
XFI_PCS_CR2	7h

### 33.6.2.9.2 Function

The XFI 10G PCS Control 2 Register contains control bits for the 10G PCS

### 33.6.2.9.3 Diagram



### 33.6.2.9.4 Fields

Field	Function
0-12 —	Reserved
13-15 PCS_TYP_SEL	PSC Type Selection PCS Type Selection Read-only field, as this PCS only supports 10GBase-R functions. Note that the value of this field does not match the 802.3 encoding for 10GBase-R.

## 33.6.2.10 XFI 10G PCS Status 2 (XFI\_PCS\_SR2)

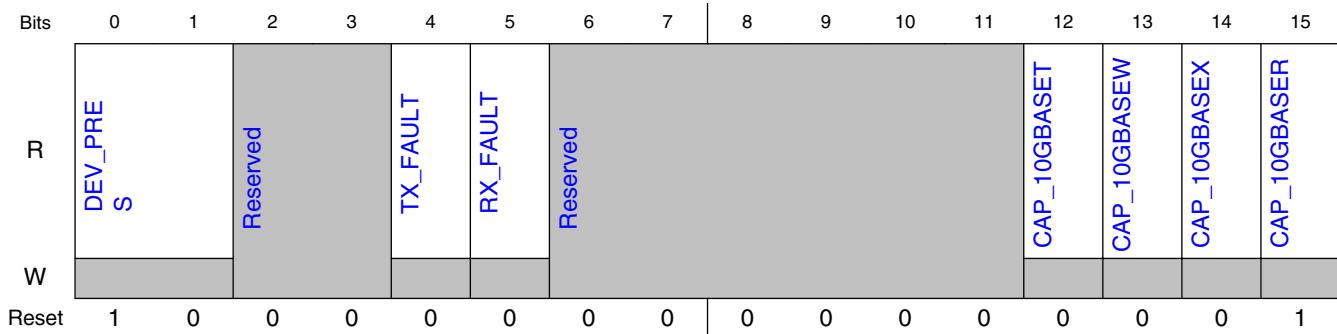
### 33.6.2.10.1 Offset

Register	Offset
XFI_PCS_SR2	8h

### 33.6.2.10.2 Function

The XFI 10G PCS Status 2 Register contains status bits for the 10G PCS

### 33.6.2.10.3 Diagram



### 33.6.2.10.4 Fields

Field	Function
0-1 DEV_PRES	Device Present Device Present 10b - device responding at this address
2-3 —	Reserved
4 TX_FAULT	Tx Fault Transmit Fault 0b - no fault condition on transmit path 1b - fault condition on transmit path
5 RX_FAULT	Rx Fault Receive Fault 0b - no fault condition on receive path 1b - fault condition on receive path
6-11 —	Reserved
12 CAP_10GBASE_T	10GBase-T Capability 10GBase-T Capable 0b - PCS is not able to support 10GBase-T PCS type
13 CAP_10GBASE_W	10GBase-W Capability 10GBase-W Capable 0b - PCS is not able to support 10GBase-W PCS type
14 CAP_10GBASE_X	10GBase-X capability 10GBase-X Capable 0b - PCS is not able to support 10GBase-X PCS type
15 CAP_10GBASE_R	10GBase-R Capability 10GBase-R Capable 1b - PCS is able to support 10GBase-R PCS type

### 33.6.2.11 XFI PCS Package Identifier Upper (XFI\_PCS\_PKG\_ID\_H)

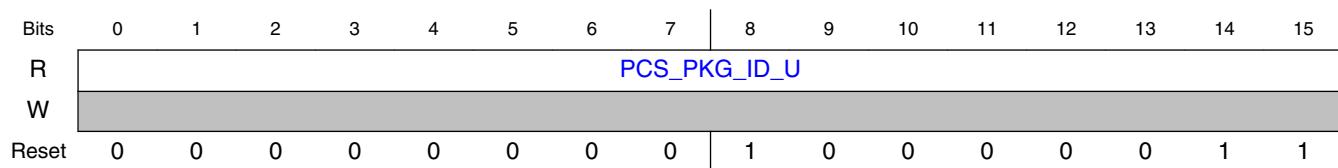
#### 33.6.2.11.1 Offset

Register	Offset
XFI_PCS_PKG_ID_H	Eh

#### 33.6.2.11.2 Function

The XFI PCS Package Identifier Upper Register contains the upper half of a 32-bit unique identifier for a particular type of package that the PCS is instantiated within

#### 33.6.2.11.3 Diagram



#### 33.6.2.11.4 Fields

Field	Function
0-15 PCS_PKG_ID_U	PCS Package ID Upper PCS Package ID Upper: 15:0 - OIU[3:18]

### 33.6.2.12 XFI PCS Package Identifier Lower (XFI\_PCS\_PKG\_ID\_L)

#### 33.6.2.12.1 Offset

Register	Offset
XFI_PCS_PKG_ID_L	Fh

### 33.6.2.12.2 Function

The XFI PCS Package Identifier Lower Register contains the lower half of a 32-bit unique identifier for a particular type of package that the PCS is instantiated within.

### 33.6.2.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									PCS_PKG_ID_L							
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.2.12.4 Fields

Field	Function
0-15 PCS_PKG_ID_L	PSC Package ID Lower PCS Package ID Lower: 15:10 - OIU[19:24] 9:4 - Manufacturer's Model Number 3:0 - Revision Number

## 33.6.2.13 XFI 10GBASE-R PCS Status 1 (XFI\_PCS\_10GR\_SR1)

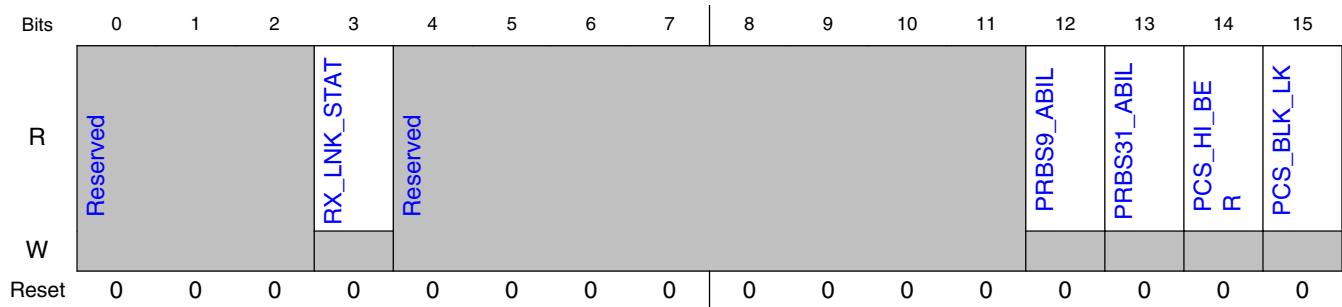
### 33.6.2.13.1 Offset

Register	Offset
XFI_PCS_10GR_SR1	20h

### 33.6.2.13.2 Function

The XFI 10GBASE-R PCS Status 1 Register contains 10GBase-R PCS status.

### 33.6.2.13.3 Diagram



### 33.6.2.13.4 Fields

Field	Function
0-2 —	Reserved
3 RX_LNK_STAT	Rx Link Status 10GBase-R Receive Link Status 0b - 10GBase-R PCS receive link is down 1b - 10GBase-R PCS receive link is up
4-11 —	Reserved
12 PRBS9_ABIL	PRBS9 Pattern Testing Ability PRBS9 Pattern Testing Ability 0b - the PCS does not support PRBS9 pattern testing
13 PRBS31_ABIL	PRBS31 Pattern testing Ability PRBS31 Pattern Testing Ability 0b - the PCS does not support PRBS31 pattern testing
14 PCS_HI_BER	PSC High BER 10GBase-R PCS high BER 0b - PCS not reporting a high BER 1b - PCS reporting a high BER
15 PCS_BLK_LK	PCS Block Lock 10GBase-R PCS block lock 0b - PCS not locked to receive blocks 1b - PCS locked to receive blocks

### 33.6.2.14 XFI 10GBASE-R PCS Status 2 (XFI\_PCS\_10GR\_SR2)

### 33.6.2.14.1 Offset

Register	Offset
XFI_PCS_10GR_SR2	21h

### 33.6.2.14.2 Function

XFI 10GBASE-R PCS Status 2 Register contains PCS status bits.

### 33.6.2.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	L_BLK_LK	LH_BER	BE_R						ERR_BLK_S							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.2.14.4 Fields

Field	Function
0 L_BLK_LK	Latched Block Lock Latched Block Lock When cleared, stays cleared until register read0: PCS does not have block lock 1b - PCS has block lock
1 LH_BER	Latched High BER Latched high BER When set, stays set until register read 0b - PCS has not reported a high BER 1b - PCS has reported a high BER
2-7 BER	BER BER Counter Does not roll over from 0x3F to 0x00 Cleared on register read or PCS reset
8-15 ERR_BLKS	Errored Blocks Counter Errored Blocks Counter Does not roll over from 0xFF to 0x00 Cleared on register read or PCS reset

### 33.6.2.15 XFI 10GBASE-R PCS Test Pattern Seed A 0 (XFI\_PCS\_TP\_SEED\_A0)

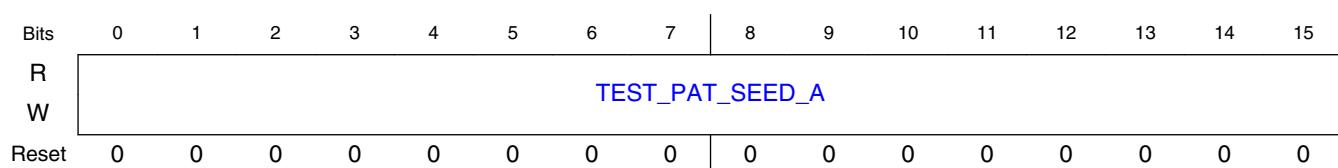
#### 33.6.2.15.1 Offset

Register	Offset
XFI_PCS_TP_SEED_A0	22h

#### 33.6.2.15.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed A Register 0 contains bits 15:0 of Test Pattern Seed A.

#### 33.6.2.15.3 Diagram



#### 33.6.2.15.4 Fields

Field	Function
0-15	Test Pattern Seed A
TEST_PAT_SEED_A	Test Pattern Seed A [15:0]

### 33.6.2.16 XFI 10GBASE-R PCS Test Pattern Seed A 1 (XFI\_PCS\_TP\_SEED\_A1)

#### 33.6.2.16.1 Offset

Register	Offset
XFI_PCS_TP_SEED_A1	23h

### 33.6.2.16.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed A Register 1 contains bits 31:16 of Test Pattern Seed A.

### 33.6.2.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TEST_PAT_SEED_A															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.2.16.4 Fields

Field	Function
0-15	Test Pattern Seed A
TEST_PAT_SEED_A	Test Pattern Seed A [31:16]

## 33.6.2.17 XFI 10GBASE-R PCS Test Pattern Seed A 2 (XFI\_PCS\_TP\_SEED\_A2)

### 33.6.2.17.1 Offset

Register	Offset
XFI_PCS_TP_SEED_A2	24h

### 33.6.2.17.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed A Register 2 contains bits 47:32 of Test Pattern Seed A.

### 33.6.2.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																

Reset 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0

### 33.6.2.17.4 Fields

Field	Function
0-15	Test Pattern Seed A
TEST_PAT_SEED_A	Test Pattern Seed A [47:32]

## 33.6.2.18 XFI 10GBASE-R PCS Test Pattern Seed A 3 (XFI\_PCS\_TP\_SEED\_A3)

### 33.6.2.18.1 Offset

Register	Offset
XFI_PCS_TP_SEED_A3	25h

### 33.6.2.18.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed A Register 3 contains bits 57:48 of Test Pattern Seed A.

### 33.6.2.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																

Reset 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0

### 33.6.2.18.4 Fields

Field	Function
0-5	Reserved
—	
6-15	Test Pattern Seed A
TEST_PAT_SE ED_A	Test Pattern Seed A [57:48]

## 33.6.2.19 XFI 10GBASE-R PCS Test Pattern Seed B 0 (XFI\_PCS\_TP\_SEED\_B0)

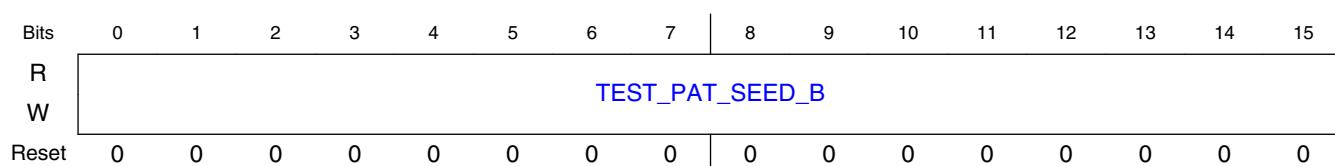
### 33.6.2.19.1 Offset

Register	Offset
XFI_PCS_TP_SEED_B0	26h

### 33.6.2.19.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed B Register 0 contains bits 15:0 of Test Pattern Seed B.

### 33.6.2.19.3 Diagram



### 33.6.2.19.4 Fields

Field	Function
0-15	Test Pattern Seed B
TEST_PAT_SE ED_B	Test Pattern Seed B [15:0]

### 33.6.2.20 XFI 10GBASE-R PCS Test Pattern Seed B 1 (XFI\_PCS\_TP\_SEED\_B1)

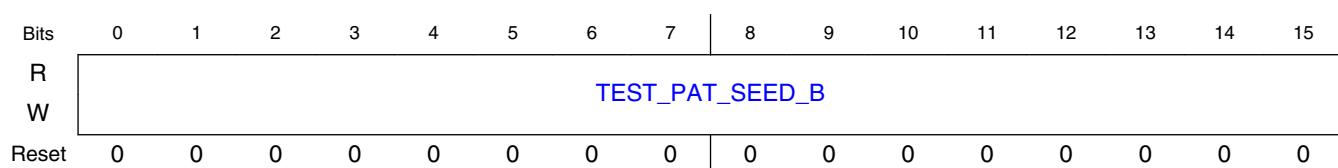
#### 33.6.2.20.1 Offset

Register	Offset
XFI_PCS_TP_SEED_B1	27h

#### 33.6.2.20.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed B Register 1 contains bits 31:16 of Test Pattern Seed B.

#### 33.6.2.20.3 Diagram



#### 33.6.2.20.4 Fields

Field	Function
0-15	Test Pattern Seed B
TEST_PAT_SEED_B	Test Pattern Seed B [31:16]

### 33.6.2.21 XFI 10GBASE-R PCS Test Pattern Seed B 2 (XFI\_PCS\_TP\_SEED\_B2)

#### 33.6.2.21.1 Offset

Register	Offset
XFI_PCS_TP_SEED_B2	28h

### 33.6.2.21.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed B Register 2 contains bits 47:32 of Test Pattern Seed B.

### 33.6.2.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									TEST_PAT_SEED_B							
W																

Reset    0    0    0    0    0    0    0    |    0    0    0    0    0    0    0    0

### 33.6.2.21.4 Fields

Field	Function
0-15	Test Pattern Seed B
TEST_PAT_SEED_B	Test Pattern Seed B [47:32]

## 33.6.2.22 XFI 10GBASE-R PCS Test Pattern Seed B 3 (XFI\_PCS\_TP\_SEED\_B3)

### 33.6.2.22.1 Offset

Register	Offset
XFI_PCS_TP_SEED_B3	29h

### 33.6.2.22.2 Function

The XFI 10GBASE-R PCS Test Pattern Seed B Register 3 contains bits 57:48 of Test Pattern Seed B.

### 33.6.2.22.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																

Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

### 33.6.2.22.4 Fields

Field	Function
0-5	Reserved
—	
6-15	Test Pattern Seed B
TEST_PAT_SE ED_B	Test Pattern Seed B [57:48]

## 33.6.2.23 XFI 10GBASE-R PCS Test Pattern Control (XFI\_PCS\_TP\_C\_R)

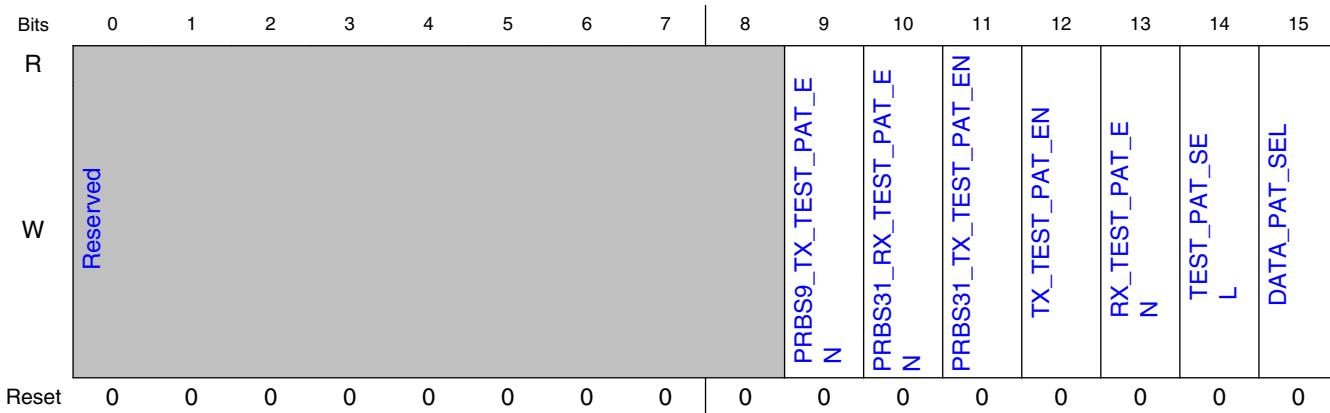
### 33.6.2.23.1 Offset

Register	Offset
XFI_PCS_TP_CR	2Ah

### 33.6.2.23.2 Function

The XFI 10GBASE-R PCS Test Pattern Control Register contains control bits for test pattern function.

### 33.6.2.23.3 Diagram



### 33.6.2.23.4 Fields

Field	Function
0-8 —	Reserved
9 PRBS9_RX_TE_ST_PAT_EN	PRBS9 Rx Test Pattern Enable PRBS9 transmit test pattern enable Read-only always set to 0 0b - Disable PRBS9 test-pattern mode on the receive path
10 PRBS31_RX_TE_ST_PAT_EN	PRBS31 Rx Test Pattern Enable PRBS31 receive test pattern enable Read-only always set to 0 0b - Disable PRBS31 test-pattern mode on the receive path
11 PRBS31_TX_TE_PAT_EN	PRBS32 Tx Test Pattern Enable PRBS31 transmit test pattern enable Read-only always set to 0 0b - Disable PRBS31 test-pattern mode on the transmit path
12 TX_TEST_PAT_EN	Tx Test Pattern Enable Transmit test pattern enable 0b - Disable transmit test pattern 1b - Enable transmit test pattern
13 RX_TEST_PAT_EN	Rx Test Pattern Enable Receive test pattern enable 0b - Disable receive test pattern 1b - Enable receive test pattern
14 TEST_PAT_SEL	Test Pattern Select Test pattern select

Table continues on the next page...

## MDIO register spaces

Field	Function
	0b - Pseudo-random test pattern 1b - Square wave test pattern
15 DATA_PAT_SE L	Data Pattern Select Transmit test pattern enable 0b - LF data pattern 1b - Zeroes pattern

### 33.6.2.24 XFI 10GBASE-R PCS Test Pattern Error Counter (XFI\_PCS\_TP\_ERR\_CNT)

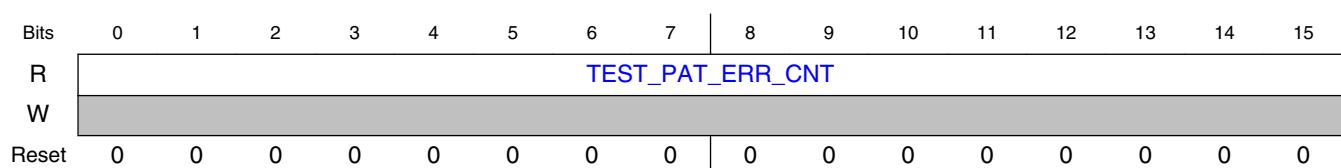
#### 33.6.2.24.1 Offset

Register	Offset
XFI_PCS_TP_ERR_CNT	2Bh

#### 33.6.2.24.2 Function

The XFI 10GBASE-R PCS Test Pattern Error Counter Register contains the error counter for the test pattern.

#### 33.6.2.24.3 Diagram



#### 33.6.2.24.4 Fields

Field	Function
0-15	Test Pattern Error Count
TEST_PAT_ER_R_CNT	Test Pattern Error Counter

### 33.6.2.25 Vendor Specific PCS Status (XFI\_PCS\_VENDOR\_SR)

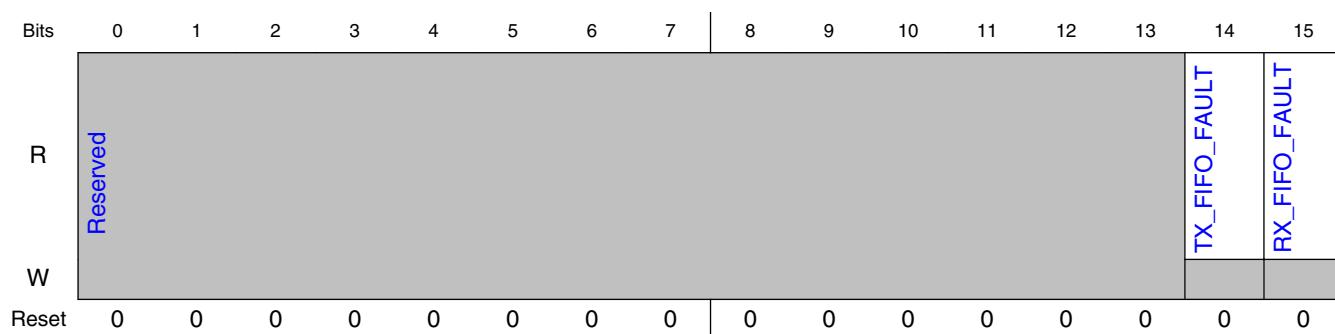
#### 33.6.2.25.1 Offset

Register	Offset
XFI_PCS_VENDOR_SR	8000h

#### 33.6.2.25.2 Function

The XFI Vendor Specific PCS Status Register contains the status for the XFI PCS.

#### 33.6.2.25.3 Diagram



#### 33.6.2.25.4 Fields

Field	Function
0-13	Reserved
14 TX_FIFO_FAULT	Tx FIFO Fault Transmit FIFO Fault Once set, stays set until register read 0b - No error on the transmit decoupling FIFO 1b - Error (overflow or underflow) on the transmit decoupling FIFO
15 RX_FIFO_FAULT	Rx FIFO Fault Receive FIFO Fault Once set, stays set until register read 0b - No error on the receive decoupling FIFO 1b - Error (overflow or underflow) on the receive decoupling FIFO

### 33.6.3 MDIO\_XFI\_AN register descriptions

The XFI auto-negotiation register space is selected when the associated XFInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 07h.

#### 33.6.3.1 MDIO\_XFI\_AN memory map

MDIO\_XFI\_AN base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	XFI AN Control (XF_AN_CR)	16	RW	0000h
1h	XFI AN Status (XF_AN_SR)	16	RO	0008h
2h	XFI AN Device Identifier Upper (XF_AN_DEV_ID_H)	16	RO	0083h
3h	XFI AN Device Identifier Lower (XF_AN_DEV_ID_L)	16	RO	E400h
5h	XFI AN Devices In Package 0 (XF_AN_DEV_PRES0)	16	RO	008Ah
6h	XFI AN Devices In Package 1 (XF_AN_DEV_PRES1)	16	RO	0000h
Eh	XFI AN Package Identifier Upper (XF_AN_PKG_ID_H)	16	RO	0083h
Fh	XFI AN Package Identifier Lower (XF_AN_PKG_ID_L)	16	RO	E400h
10h	XFI AN Advertisement 0 (XF_AN_ADVERT0)	16	RW	0001h
11h	XFI AN Advertisement 1 (XF_AN_ADVERT1)	16	RW	001Fh
12h	XFI AN Advertisement 2 (XF_AN_ADVERT2)	16	RW	C000h
13h	XFI AN LP Base Page Ability 0 (XF_AN_LP_BASE_PG_ABIL0)	16	RO	0000h
14h	XFI AN LP Base Page Ability 1 (XF_AN_LP_BASE_PG_ABIL1)	16	RO	0000h
15h	XFI AN LP Base Page Ability 2 (XF_AN_LP_BASE_PG_ABIL2)	16	RO	0000h
16h	XFI AN Extended Next Page Transmit 0 (XF_AN_XNP_TX0)	16	RW	2001h
17h	XFI AN Extended Next Page Transmit 1 (XF_AN_XNP_TX1)	16	RW	0000h
18h	XFI AN Extended Next Page Transmit 2 (XF_AN_XNP_TX2)	16	RW	0000h
19h	XFI AN LP Extended Next Page Ability 0 (XF_AN_LP_XNP_ABIL0)	16	RO	0000h
1Ah	XFI AN LP Extended Next Page Ability 1 (XF_AN_LP_XNP_ABIL1)	16	RO	0000h
1Bh	XFI AN LP Extended Next Page Ability 2 (XF_AN_LP_XNP_ABIL2)	16	RO	0000h
30h	XFI Backplane Ethernet Status (XF_BP_STAT)	16	RO	0000h

#### 33.6.3.2 XFI AN Control (XF\_AN\_CR)

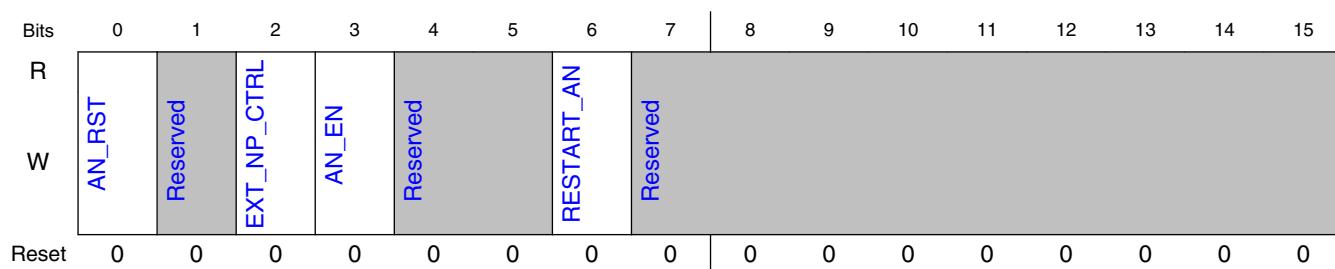
### 33.6.3.2.1 Offset

Register	Offset
XF_AN_CR	0h

### 33.6.3.2.2 Function

The AN Control Register contains general controls for auto-negotiation.

### 33.6.3.2.3 Diagram



### 33.6.3.2.4 Fields

Field	Function
0 AN_RST	AN Reset AN reset Self-clearing bit 0b - AN normal operation 1b - AN reset
1 —	Reserved
2 EXT_NP_CTRL	Extended Next Page Control Extended Next Page Control 0b - Extended next pages are disabled 1b - Extended next pages are enabled
3 AN_EN	AN Enable Auto-negotiation enable 0b - Disable auto-negotiation process 1b - Enable auto-negotiation process
4-5 —	Reserved
6	Restart AN

Table continues on the next page...

## MDIO register spaces

Field	Function
RESTART_AN	Restart auto-negotiation 0b - Auto-negotiation in process, disabled, or not supported 1b - Restart auto-negotiation process
7-15 —	Reserved

### 33.6.3.3 XFI AN Status (XF\_AN\_SR)

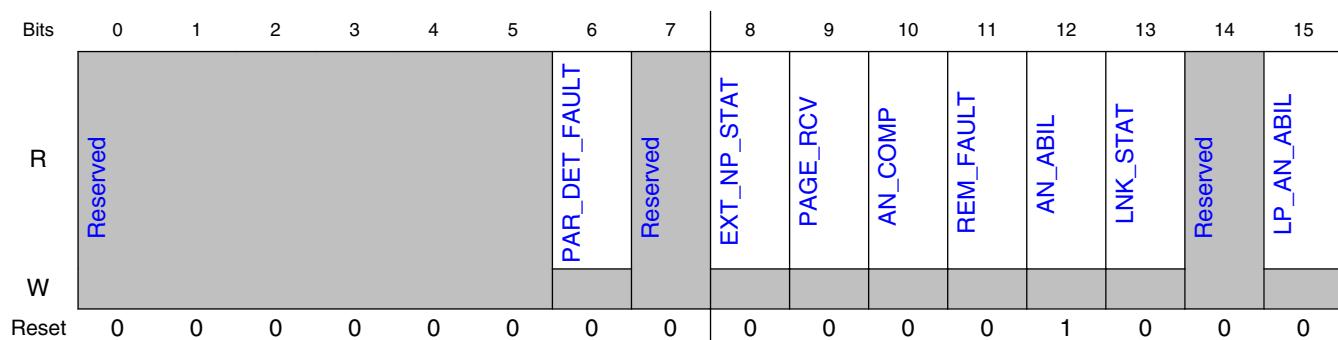
#### 33.6.3.3.1 Offset

Register	Offset
XF_AN_SR	1h

#### 33.6.3.3.2 Function

The XFI AN Status Register contains status bits for the XFI auto-negotiate function.

#### 33.6.3.3.3 Diagram



#### 33.6.3.3.4 Fields

Field	Function
0-5 —	Reserved
6	Parallel Detection Fault Parallel detection fault

Table continues on the next page...

Field	Function
PAR_DET_FAU_LT	Once set, stays set until register read 0b - A fault has not been detected via the parallel detection function 1b - A fault has been detected via the parallel detection function
7 —	Reserved
8 EXT_NP_STAT	Extended Next Page Status Extended next page status 0b - Extended next page is not allowed 1b - Extended next page format is used
9 PAGE_RCV	Page Received Page received Once set, stays set until register read 0b - A page has not been received 1b - A page has been received
10 AN_COMP	AN Complete Auto-Negotiation complete 0b - Auto-negotiation process not completed 1b - Auto-negotiation process completed
11 REM_FAULT	remote fault Remote Fault Once set, stays set until register read. 0b - No remote fault condition detected 1b - Remote fault condition detected
12 AN_ABIL	AN ability Auto-Negotiation Ability 1b - PHY is able to perform Auto-Negotiation
13 LNK_STAT	Link Status Link Status Once set, stays set until register read. 0b - Link is down 1b - Link is up
14 —	Reserved
15 LP_AN_ABIL	Link Partner AN ability Link partner Auto-Negotiation ability 0b - LP is not able to perform Auto-Negotiation 1b - LP is able to perform Auto-Negotiation

### 33.6.3.4 XFI AN Device Identifier Upper (XFI\_AN\_DEV\_ID\_H)

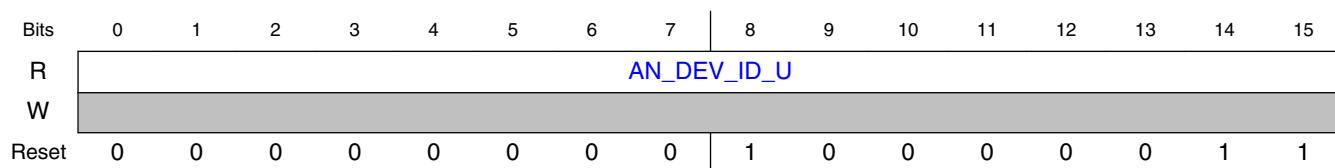
#### 33.6.3.4.1 Offset

Register	Offset
XFI_AN_DEV_ID_H	2h

#### 33.6.3.4.2 Function

The AN Device Identifier Upper Register contains the upper half of the 32-bit device identifier.

#### 33.6.3.4.3 Diagram



#### 33.6.3.4.4 Fields

Field	Function
0-15	AN Device ID Upper
AN_DEV_ID_U	AN Device ID Upper: 15:0 - OIU[3:18]

### 33.6.3.5 XFI AN Device Identifier Lower (XFI\_DEV\_ID\_L)

#### 33.6.3.5.1 Offset

Register	Offset
XFI_DEV_ID_L	3h

### 33.6.3.5.2 Function

The AN Device Identifier Lower Register contains the lower half of the 32-bit device identifier.

### 33.6.3.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	AN_DEV_ID_L															
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.3.5.4 Fields

Field	Function
0-15 AN_DEV_ID_L	AN Device ID Lower  AN Device ID Lower: 15:10 - OIU[19:24] 9:4 - Manufacturer's Model Number 3:0 - Revision Number

## 33.6.3.6 XFI AN Devices In Package 0 (XFI\_AN\_DEV\_PRES0)

### 33.6.3.6.1 Offset

Register	Offset
XFI_AN_DEV_PRES0	5h

### 33.6.3.6.2 Function

The XFI AN Devices in Package 0 Register contains half of the AN devices in package status.

### 33.6.3.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								AN_PRE_S	TC_PRE_S	DTE_XS_PRE_S	PHY_XS_PRE_S	PCS_PRE_S	WIS_PRE_S	PMD_PMA_PRES	C22_REG_PRE_S
W	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0

### 33.6.3.6.4 Fields

Field	Function	
0-7 —	Reserved	
8 AN_PRES	AN Present Auto-negotiation Present 1b - the PCS implements the Auto-Negotiation function	
9 TC_PRES	TC Present TC Present 0b - the PCS does not implement TC functions	
10 DTE_XS_PRES	DTE XS Present DTE XS Present 0b - the PCS does not implement DTE XS functions	
11 PHY_XS_PRES	PHY XS Present PHY XS Present 0b - the PCS does not implement PHY XS functions.	
12 PCS_PRES	PCS Present PCS Present 1b - the PCS implements PCS functions	
13 WIS_PRES	WIS Present WIS Present 0b - the PCS does not implement a WIS function.	
14 PMD_PMA_PRES	PMD/PMA Present PMD/PMA Present 1b - the PCS implements PMD/PMA functions (Link Training and FEC functions)	
15 C22_REG_PRE_S	Clause 22 Registers Present Clause 22 Registers Present 0b - the PCS does not implement the Clause 22 Registers.	

### 33.6.3.7 XFI AN Devices In Package 1 (XFI\_AN\_DEV\_PRES1)

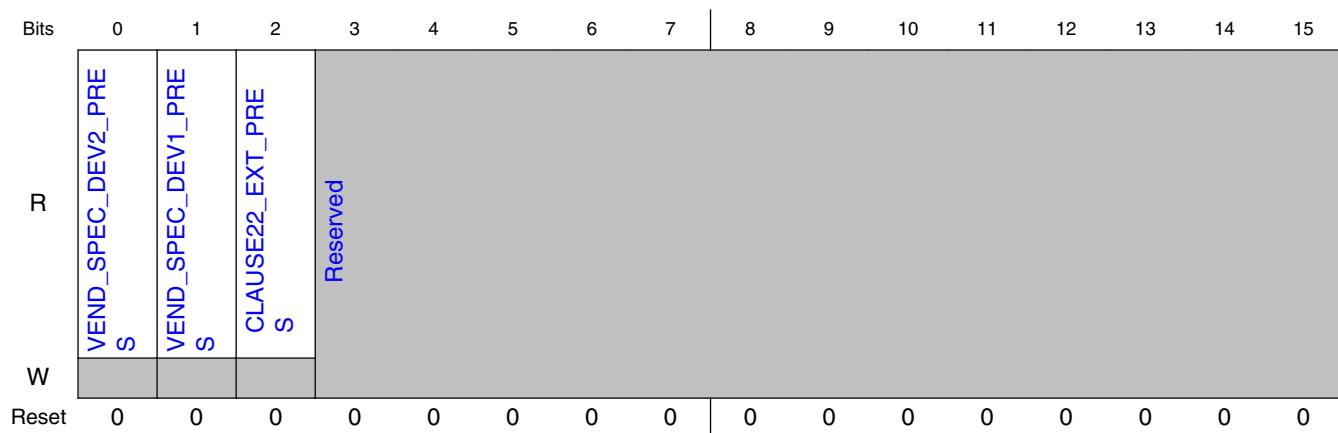
#### 33.6.3.7.1 Offset

Register	Offset
XFI_AN_DEV_PRES1	6h

#### 33.6.3.7.2 Function

The XFI AN Devices in Package 1 Register contains half of the AN devices in package status.

#### 33.6.3.7.3 Diagram



#### 33.6.3.7.4 Fields

Field	Function
0 VEND_SPEC_DEV2_PRES	Vendor Specific Device 2 Present Vendor Specific Device 2 Present 0b - the PCS does not implement any vendor specific device.
1 VEND_SPEC_DEV1_PRES	Vendor Specific Device 1 Present Vendor Specific Device 1 Present 0b - the PCS does not implement any vendor specific device.
2	Clause 22 Extension Present

Table continues on the next page...

## MDIO register spaces

Field	Function
CLAUSE22_EX_T_PRES	Clause 22 Extension Present 0b - the PCS does not implement Clause 22 extensions.
3-15 —	Reserved

### 33.6.3.8 XFI AN Package Identifier Upper (XF\_AN\_PKG\_ID\_H)

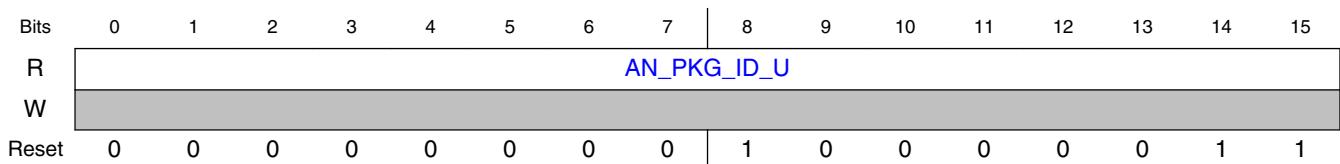
#### 33.6.3.8.1 Offset

Register	Offset
XF_AN_PKG_ID_H	Eh

#### 33.6.3.8.2 Function

The AN Package Identifier Upper Register contains the upper half of the 32-bit package identifier.

#### 33.6.3.8.3 Diagram



#### 33.6.3.8.4 Fields

Field	Function
0-15 AN_PKG_ID_U	AN Package ID Upper: 15:0 - OIU[3:18]

### 33.6.3.9 XFI AN Package Identifier Lower (XF\_AN\_PKG\_ID\_L)

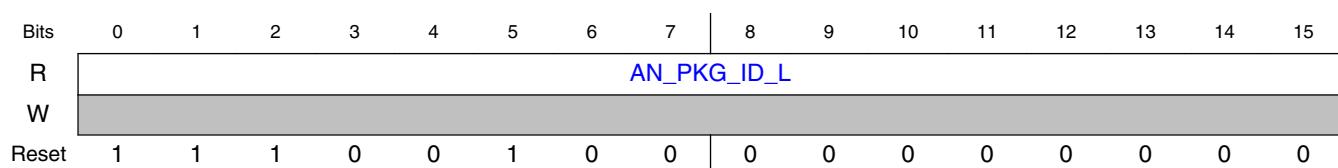
### 33.6.3.9.1 Offset

Register	Offset
XFI_AN_PKG_ID_L	Fh

### 33.6.3.9.2 Function

The AN Package Identifier Lower Register contains the lower half of the 32-bit package identifier.

### 33.6.3.9.3 Diagram



### 33.6.3.9.4 Fields

Field	Function
0-15 AN_PKG_ID_L	AN Package ID Lower: 15:10 - OIU[19:24] 9:4 - Manufacturer's Model Number 3:0 - Revision Number

## 33.6.3.10 XFI AN Advertisement 0 (XFI\_AN\_ADVERT0)

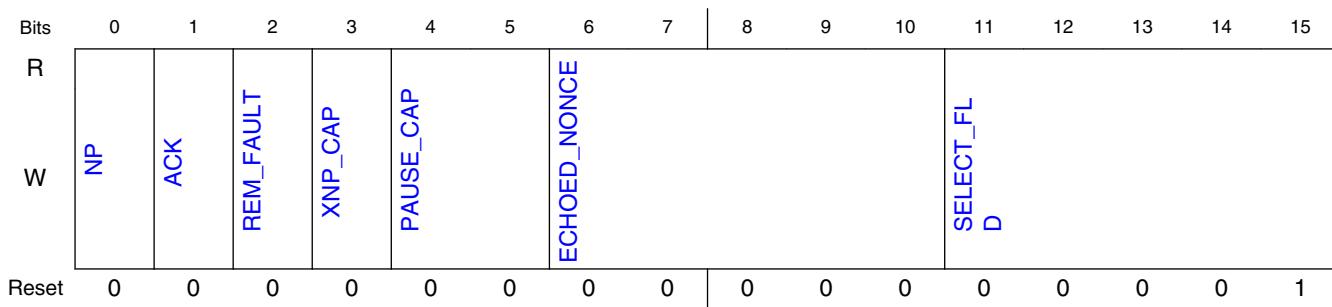
### 33.6.3.10.1 Offset

Register	Offset
XFI_AN_ADVERT0	10h

### 33.6.3.10.2 Function

The XFI AN Advertisement Register 0 contains bits 15:0 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2, where register 2 holds the most significant 16 bits of value.

### 33.6.3.10.3 Diagram



### 33.6.3.10.4 Fields

Field	Function
0 NP	Next Page Next Page: 0b - Device does not have a Next Page to send 1b - Device has a Next Page to send
1 ACK	Ack Should always be set to 0
2 REM_FAULT	Remote Fault Remote Fault 0b - Normal operation 1b - Device is indicating a remote fault condition
3 XNP_CAP	Extended Next Page capability Extended Next Page Capability 0b - Device does not support extended next pages
4-5 PAUSE_CAP	Pause Capability Pause Capability (ASM_DIR:PAUSE) 00b - No PAUSE 01b - Symmetric PAUSE 10b - Asymmetric PAUSE toward link partner 11b - Both Symmetric PAUSE and Asymmetric PAUSE toward local device
6-10	Echoed Nonce Echoed Nonce

Table continues on the next page...

Field	Function
ECHOED_NON_CE	Nonce value received from link partner
11-15 SELECT_FLD	Selector Field Selector Field 00001b - IEEE Standard 802.3

### 33.6.3.11 XFI AN Advertisement 1 (XFI\_AN\_ADVERT1)

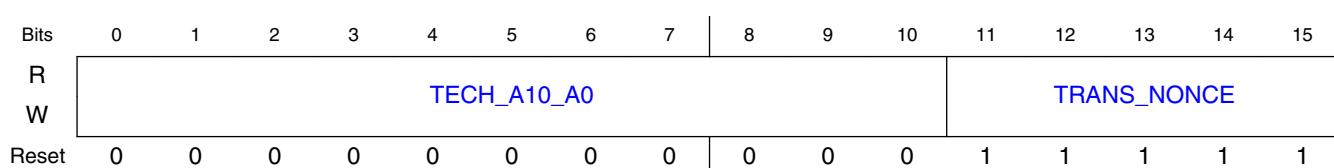
#### 33.6.3.11.1 Offset

Register	Offset
XFI_AN_ADVERT1	11h

#### 33.6.3.11.2 Function

The XFI AN Advertisement Register 1 contains bits 31:16 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2, where register 2 holds the most significant 16 bits of value.

#### 33.6.3.11.3 Diagram



#### 33.6.3.11.4 Fields

Field	Function
0-10 TECH_A10_A0	Technology A10:A0 Technology Field (A10:A0) A10:A0: reserved must be set to 0
11-15	Transmitted Nonce Transmitted Nonce

## MDIO register spaces

Field	Function
TRANS_NONCE	Must be set to unique value per device prior to enabling auto-negotiation Defaults to 0h1F

### 33.6.3.12 XFI AN Advertisement 2 (XFI\_AN\_ADVERT2)

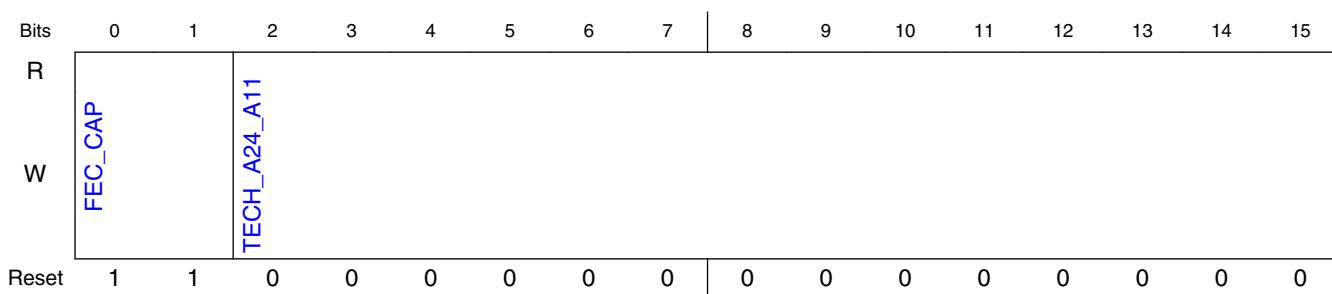
#### 33.6.3.12.1 Offset

Register	Offset
XFI_AN_ADVERT2	12h

#### 33.6.3.12.2 Function

The XFI AN Advertisement Register 2 contains bits 47:32 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2, where register 2 holds the most significant 16 bits of value.

#### 33.6.3.12.3 Diagram



#### 33.6.3.12.4 Fields

Field	Function
0-1 FEC_CAP	FEC Capability 00b - Device is not capable of or requesting FEC 11b - Device is capable of and requesting FEC support
2-15	Technology A24:A11

Field	Function
TECH_A24_A11	Technology Field (A24:A11) A24:A11: reserved must be set to 0

### 33.6.3.13 XFI AN LP Base Page Ability 0 (XFI\_AN\_LP\_BASE\_PG\_ABIL0)

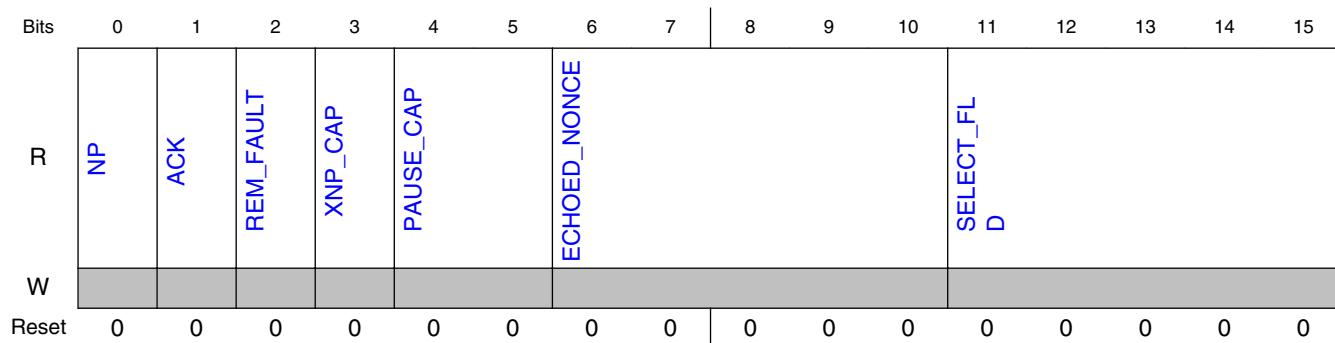
#### 33.6.3.13.1 Offset

Register	Offset
XFI_AN_LP_BASE_PG_ABIL0	13h

#### 33.6.3.13.2 Function

The XFI AN LP Base Page Ability Register 0 contains bits 15:0 of the link partner base storage. The 48-bit value is stored over registers 0, 1 and 2, where register 2 holds the most significant 16 bits of the value. The contents of the registers are identical to the XFI AN advertisement registers (see above).

#### 33.6.3.13.3 Diagram



#### 33.6.3.13.4 Fields

Field	Function
0	Next Page
NP	Next Page:

Table continues on the next page...

## MDIO register spaces

Field	Function
	0b - Link Partner does not have a Next Page to send 1b - Link Partner has a Next Page to send
1 ACK	Ack Acknowledge 0b - Base Ability data not valid 1b - Base Ability data valid
2 REM_FAULT	Remote Fault Remote Fault 0b - Normal operation 1b - Link Partner is indicating a remote fault condition
3 XNP_CAP	Extended Next Page Capability Extended Next Page Capability 0b - Link Partner does not support extended next pages 1b - Link Partner supports extended next pages
4-5 PAUSE_CAP	Pause Capability Pause Capability (ASM_DIR:PAUSE) 00b - No PAUSE 01b - Symmetric PAUSE 10b - Asymmetric PAUSE toward local device 11b - Both Symmetric PAUSE and Asymmetric PAUSE toward link partner
6-10 ECHOED_NONCE	Echoed Nonce Echoed Nonce Nonce value received by link partner
11-15 SELECT_FLD	Selector Field Selector Field 00001b - IEEE Standard 802.3

### 33.6.3.14 XFI AN LP Base Page Ability 1 (XFI\_AN\_LP\_BASE\_PG\_ABIL1)

#### 33.6.3.14.1 Offset

Register	Offset
XFI_AN_LP_BASE_PG_ABIL1	14h

### 33.6.3.14.2 Function

The XFI AN LP Base Page Ability Register 1 contains bits 31:16 of the link partner base storage. The 48-bit value is stored over registers 0, 1 and 2, where register 2 holds the most significant 16 bits of the value. The contents of the registers are identical to the XFI AN advertisement registers (see above).

### 33.6.3.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TECH_A10_A0								TRANS_NONCE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.3.14.4 Fields

Field	Function
0-10	Technology A10:A0
TECH_A10_A0	Technology Field (A10:A0) A10:A0: reserved must be set to 0
11-15	Transmitted Nonce
TRANS_NONC E	Transmitted Nonce Must be set to unique value per device prior to enabling auto-negotiation

## 33.6.3.15 XFI AN LP Base Page Ability 2 (XFI\_AN\_LP\_BASE\_PG\_ABIL2)

### 33.6.3.15.1 Offset

Register	Offset
XFI_AN_LP_BASE_PG_ABIL2	15h

### 33.6.3.15.2 Function

The XFI AN LP Base Page Ability Register 2 contains bits 47:31 of the link partner base storage. The 48-bit value is stored over registers 0, 1 and 2, where register 2 holds the most significant 16 bits of the value. The contents of the registers are identical to the XFI AN advertisement registers (see above).

### 33.6.3.15.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	FEC_CAP		TECH_A24_A11													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.3.15.4 Fields

Field	Function
0-1 FEC_CAP	FEC Capability FEC Capability (F1:F0) 00b - Link partner is not capable of or requesting FEC 01b - Link partner is capable of but not requesting FEC 11b - Link partner is capable of and requesting FEC
2-15 TECH_A24_A11	Technology A24:A11 Technology Field (A24:A11) A24:A11: reserved must be set to 0

## 33.6.3.16 XFI AN Extended Next Page Transmit 0 (XFI\_AN\_XNP\_TX0)

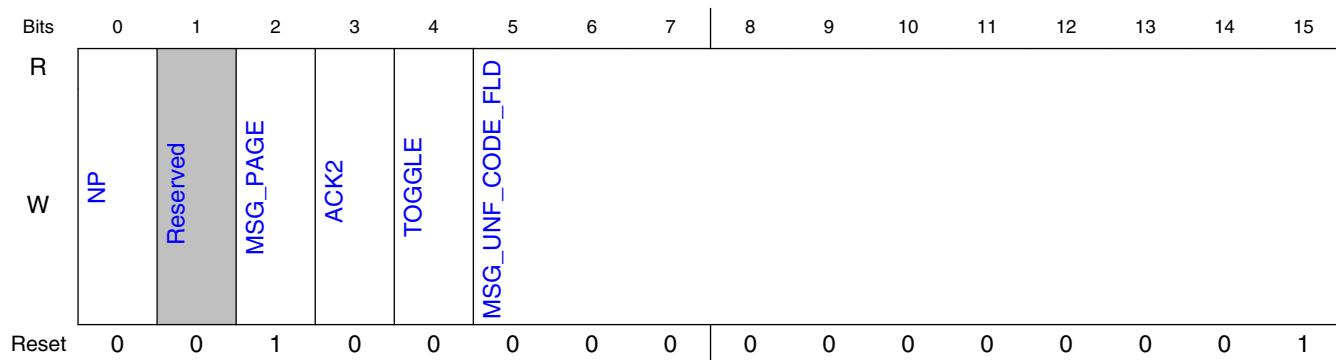
### 33.6.3.16.1 Offset

Register	Offset
XFI_AN_XNP_TX0	16h

### 33.6.3.16.2 Function

The XFI AN XNP Transmit Register 0 contains bits 15:0 of the extended next page transmit data.

### 33.6.3.16.3 Diagram



### 33.6.3.16.4 Fields

Field	Function
0 NP	Next Page Next Page 0b - Device does not have any more Next Pages to send 1b - Device has more Next Pages to Send
1 —	Reserved
2 MSG_PAGE	Message Page Message Page 0b - Next page is an unformatted page 1b - Next page is a message page
3 ACK2	Ack 2 Acknowledge 2 0b - The receiver is not able to act on the information defined in the message 1b - The receiver is able to act on the information defined in the message
4 TOGGLE	Toggle Toggle
5-15 MSG_UNF_CODE_FLD	Message/Unformatted Code Field Message/Unformatted Code Field When MSG_PAGE=1: Message code field, else unformatted code field. For the null message code, the value is 0x1

### 33.6.3.17 XFI AN Extended Next Page Transmit 1 (XFI\_AN\_XNP\_TX1)

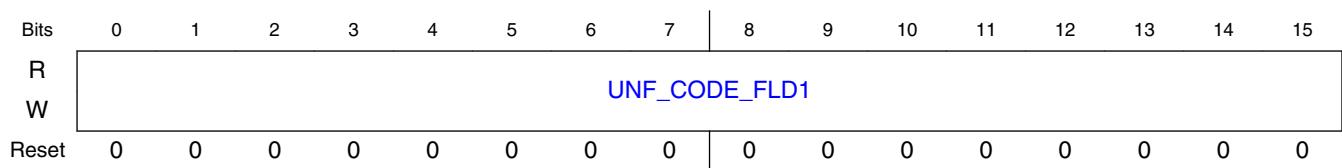
#### 33.6.3.17.1 Offset

Register	Offset
XFI_AN_XNP_TX1	17h

#### 33.6.3.17.2 Function

The XFI AN XNP Transmit Register 1 contains bits 31:16 of the extended next page transmit data.

#### 33.6.3.17.3 Diagram



#### 33.6.3.17.4 Fields

Field	Function
0-15	Unformatted Code Field 1
UNF_CODE_FL D1	Unformatted Code Field 1

### 33.6.3.18 XFI AN Extended Next Page Transmit 2 (XFI\_AN\_XNP\_TX2)

#### 33.6.3.18.1 Offset

Register	Offset
XFI_AN_XNP_TX2	18h

### 33.6.3.18.2 Function

The XFI AN XNP Transmit Register 2 contains bits 47:32 of the extended next page transmit data.

### 33.6.3.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W	UNF_CODE_FLD2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.3.18.4 Fields

Field	Function
0-15	Unformatted Code Field 2
UNF_CODE_FL D2	Unformatted Code Field 2

## 33.6.3.19 XFI AN LP Extended Next Page Ability 0 (XFI\_AN\_LP\_XNP\_ABIL0)

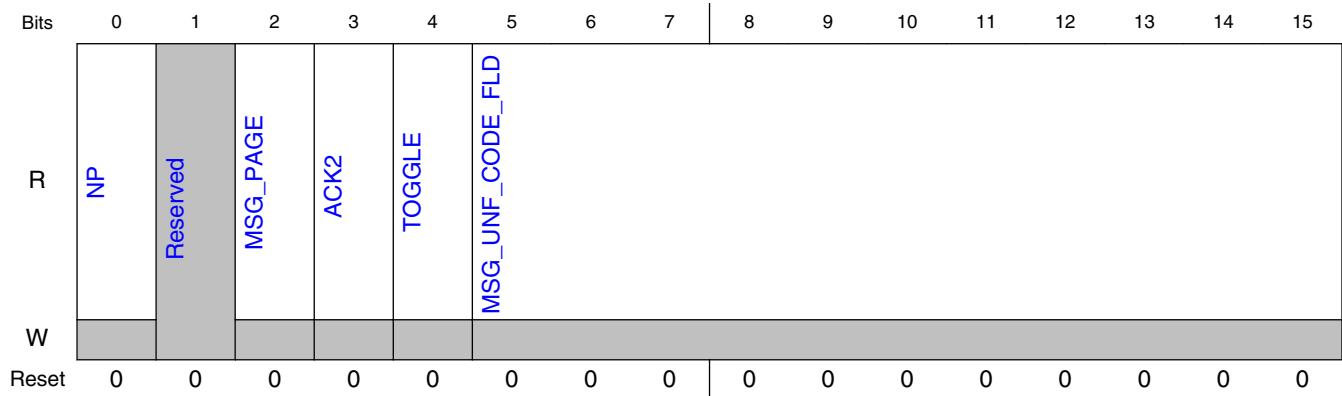
### 33.6.3.19.1 Offset

Register	Offset
XFI_AN_LP_XNP_ABIL0	19h

### 33.6.3.19.2 Function

The XFI AN LP XNP Ability Register 0 contains bits 15:0 of the extended next page value from the remote device.

### 33.6.3.19.3 Diagram



### 33.6.3.19.4 Fields

Field	Function
0 NP	Next Page Next Page 0b - Device does not have any more Next Pages to send 1b - Device has more Next Pages to Send
1 —	Reserved
2 MSG_PAGE	Message Page Message Page 0b - Next page is an unformatted page 1b - Next page is a message page
3 ACK2	Ack 2 Acknowledge 2 0b - The receiver is not able to act on the information defined in the message 1b - The receiver is able to act on the information defined in the message
4 TOGGLE	Toggle Toggle
5-15 MSG_UNF_CODE_FLD	Message/Unformatted Code Field Message/Unformatted Code Field When MSG_PAGE=1: Message code field, else unformatted code field. For the null message code, the value is 0x1

### 33.6.3.20 XFI AN LP Extended Next Page Ability 1 (XFI\_AN\_LP\_XNP\_ABIL1)

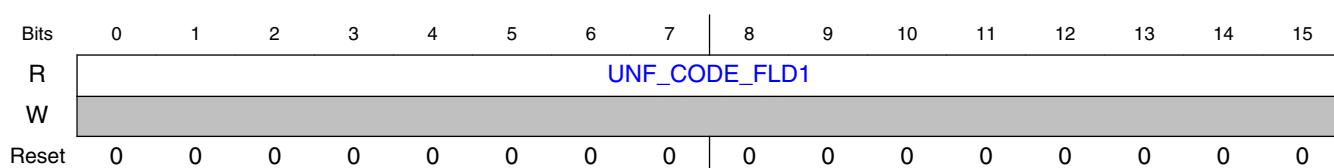
#### 33.6.3.20.1 Offset

Register	Offset
XFI_AN_LP_XNP_ABIL1	1Ah

#### 33.6.3.20.2 Function

The XFI AN LP XNP Ability Register 1 contains bits 31:16 of the extended next page value from the remote device.

#### 33.6.3.20.3 Diagram



#### 33.6.3.20.4 Fields

Field	Function
0-15	Unformatted Code Field 1
UNF_CODE_FLD1	Unformatted Code Field 1

### 33.6.3.21 XFI AN LP Extended Next Page Ability 2 (XFI\_AN\_LP\_XNP\_ABIL2)

#### 33.6.3.21.1 Offset

Register	Offset
XFI_AN_LP_XNP_ABIL2	1Bh

### 33.6.3.21.2 Function

The XFI AN LP XNP Ability Register 2 contains bits 47:32 of the extended next page value from the remote device.

### 33.6.3.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									UNF_CODE_FLD2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.3.21.4 Fields

Field	Function
0-15	Unformatted Code Field 2
UNF_CODE_FL D2	Unformatted Code Field 2

## 33.6.3.22 XFI Backplane Ethernet Status (XFI\_BP\_STAT)

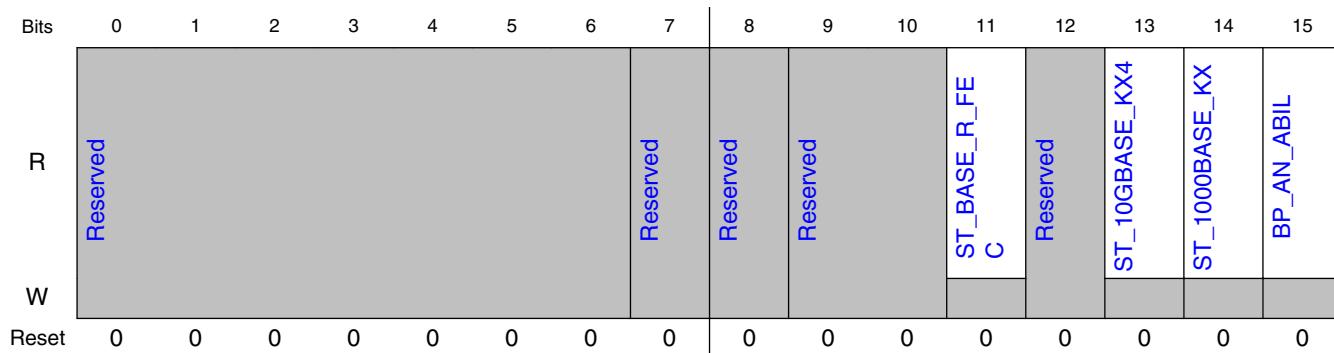
### 33.6.3.22.1 Offset

Register	Offset
XFI_BP_STAT	30h

### 33.6.3.22.2 Function

The XFI Backplane Ethernet Status Register provides the result after auto-negotiation.

### 33.6.3.22.3 Diagram



### 33.6.3.22.4 Fields

Field	Function
0-6 —	Reserved
7 —	Reserved
8 —	Reserved
9-10 —	Reserved
11 ST_BASE_R_FEC	Base-R FEC If 1, negotiated to perform Base-R FEC
12 —	Reserved
13 ST_10GBASE_KX4	10GBase-KX4 Always 0 after negotiation complete
14 ST_1000BASE_KX	1000Base-KX Always 0 after negotiation complete
15 BP_AN_ABIL	Backplane AN Ability Backplane Base-R capable PHY type is implemented Always 1

## 33.6.4 MDIO\_XFI\_VENDOR\_SPEC register descriptions

The XFI vendor-specific 1 register space is selected when the associated XFIInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 1Eh.

### 33.6.4.1 MDIO\_XFI\_VENDOR\_SPEC memory map

MDIO\_XFI\_VENDOR\_SPEC base address: 0h

Offset	Register	Width (in bits)	Access	Reset value
0h	XFI Revision (XFI_VND_REV)	16	RO	1010h
1h	XFI Scratch (XFI_VND_SCRATCH)	16	RW	0000h
2h	XFI PCS Interrupt Event (XFI_VND_PCS_INT)	16	RO	0000h
3h	XFI PCS Interrupt Mask (XFI_VND_PCS_INT_MSK)	16	RW	0000h
4h	XFI Auto-Negotiation Interrupt Event (XFI_VND_AN_INT)	16	RO	0000h
5h	XFI Auto-Negotiation Interrupt Mask (XFI_VND_AN_INT_MSK)	16	RW	0000h
6h	XFI Link Training Interrupt Event (XFI_VND_LT_INT)	16	RO	0000h
7h	XFI Link Training Interrupt Mask (XFI_VND_LT_INT_MSK)	16	RW	0000h

### 33.6.4.2 XFI Revision (XFI\_VND\_REV)

#### 33.6.4.2.1 Offset

Register	Offset
XFI_VND_REV	0h

#### 33.6.4.2.2 Function

The XFI revision register contains revision information for the XFI PCS.

### 33.6.4.2.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R					CFG_VER				IP_VER			IP_REV				
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0

### 33.6.4.2.4 Fields

Field	Function
0-7	Configuration Version
CFG_VER	Integration Version
8-11	IP Version
IP_VER	IP Version
12-15	IP Revision
IP_REV	IP Revision

### 33.6.4.3 XFI Scratch (XFI\_VND\_SCRATCH)

#### 33.6.4.3.1 Offset

Register	Offset
XFI_VND_SCRATCH	1h

#### 33.6.4.3.2 Function

The XFI scratch register contains a read/writeable scratch register that can be used to test MDIO register access.

#### 33.6.4.3.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									SCRATCH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.4.3.4 Fields

Field	Function
0-15	Scratch register
SCRATCH	Scratch register

### 33.6.4.4 XFI PCS Interrupt Event (XFI\_VND\_PCS\_INT)

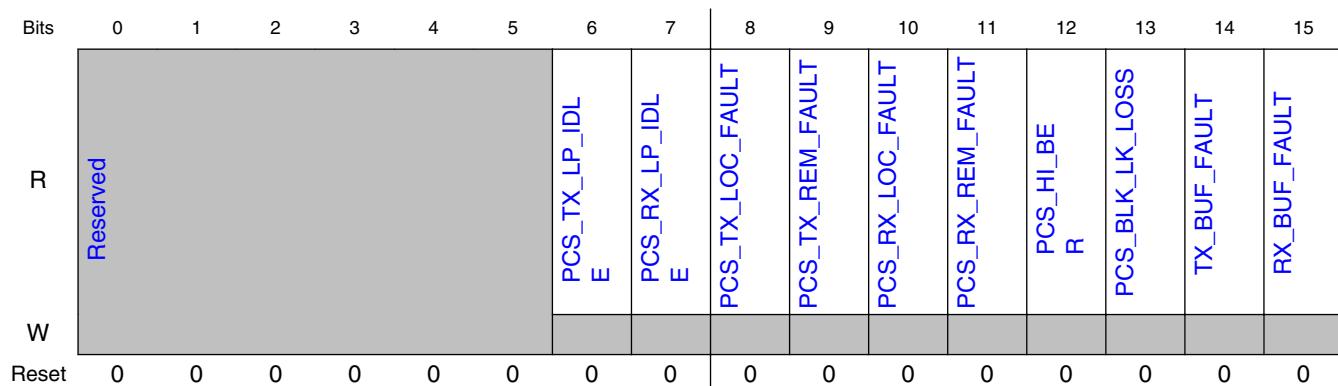
#### 33.6.4.4.1 Offset

Register	Offset
XFI_VND_PCS_INT	2h

#### 33.6.4.4.2 Function

The XFI PCS interrupt event register contains interrupt event bits for PCS function. Note that XFI PCS interrupt event register bits are cleared when read.

#### 33.6.4.4.3 Diagram



#### 33.6.4.4.4 Fields

Field	Function
0-5	Reserved
6	PCS Tx Low Power Idle

Table continues on the next page...

Field	Function
PCS_TX_LP_ID LE	PCS Tx Low Power Idle Cleared on register read 0b - PCS transmit path has not detected a Low Power Idle condition 1b - PCS transmit path has detected a Low Power Idle condition
7 PCS_RX_LP_ID LE	PCS Rx Low Power Idle PCS Rx Low Power Idle Cleared on register read 0b - PCS receive path has not detected a Low Power Idle condition 1b - PCS receive path has detected a Low Power Idle condition
8 PCS_TX_LOC_ FAULT	PCS Tx Local Fault PCS Tx Local Fault Cleared on register read 0b - PCS transmit path has not detected a Local Fault condition 1b - PCS transmit path has detected a Local Fault condition
9 PCS_TX_Rem_ FAULT	PCS Tx Remote Fault PCS Tx Remote Fault Cleared on register read 0b - PCS transmit path has not detected a Remote Fault condition 1b - PCS transmit path has detected a Remote Fault condition
10 PCS_RX_LOC_ FAULT	PCS Rx Local Fault PCS Rx Local Fault Cleared on register read 0b - PCS receive path has not detected a Local Fault condition 1b - PCS receive path has detected a Local Fault condition
11 PCS_RX_Rem_ FAULT	PCS Rx Remote Fault PCS Rx Remote Fault Cleared on register read 0b - PCS receive path has not detected a Remote Fault condition 1b - PCS receive path has detected a Remote Fault condition
12 PCS_HI_BER	PCS High BER PCS High BER Cleared on register read 0b - PCS receive path has not detected a High Bit Error Rate condition 1b - PCS receive path has detected a High Bit Error Rate condition
13 PCS_BLK_LK_L OSS	PCS Block Lock Loss PCS Block Lock Loss Cleared on register read 0b - PCS block lock has not lost synchronization 1b - PCS block lock has lost synchronization
14 TX_BUF_FAUL T	Tx Buffer Fault Tx Buffer Fault Cleared on register read

Table continues on the next page...

## MDIO register spaces

Field	Function
	0b - No buffer error detected on Transmit Rate Matching FIFO 1b - Buffer error detected on Transmit Rate Matching FIFO
15 RX_BUF_FAUL T	Rx Buffer Fault  Rx Buffer Fault  Cleared on register read  0b - No buffer error detected on Receive Rate Matching FIFO 1b - Buffer error detected on Receive Rate Matching FIFO

### 33.6.4.5 XFI PCS Interrupt Mask (XFI\_VND\_PCS\_INT\_MSK)

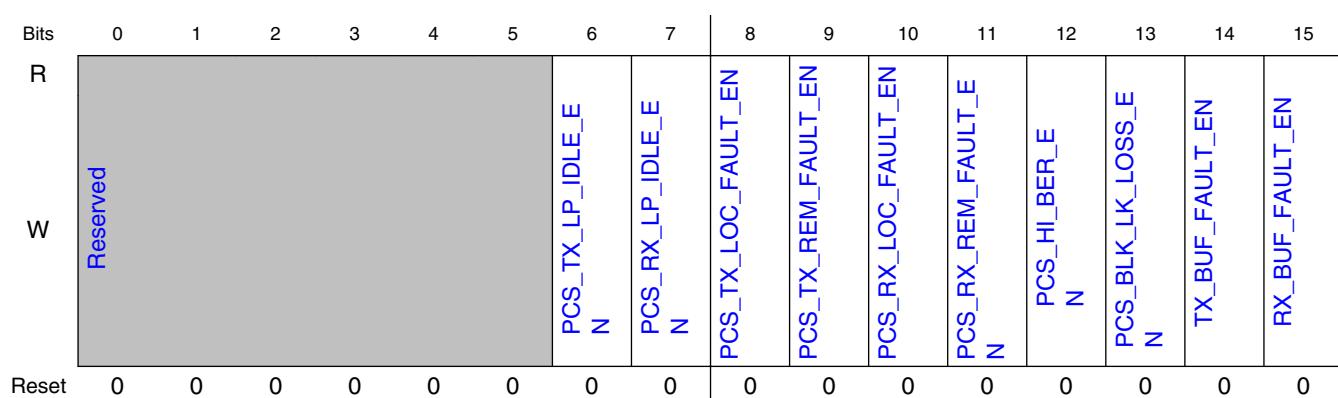
#### 33.6.4.5.1 Offset

Register	Offset
XFI_VND_PCS_INT_MSK	3h

#### 33.6.4.5.2 Function

The XFI PCS Interrupt Mask Register contains the interrupt mask bits for the above PCS interrupt event bits.

#### 33.6.4.5.3 Diagram



### 33.6.4.5.4 Fields

Field	Function
0-5 —	Reserved
6 PCS_TX_LP_ID LE_EN	PCS Tx Low Power Idle Interrupt Mask PCS Tx Low Power Idle Interrupt Mask 0b - PCS Tx Low Power Idle events do not cause an interrupt 1b - PCS Tx Low Power Idle events cause an interrupt
7 PCS_RX_LP_ID LE_EN	PCS Rx Low Power Idle PCS Rx Low Power Idle 0b - PCS Rx Low Power Idle events do not cause an interrupt 1b - PCS Rx Low Power Idle events cause an interrupt
8 PCS_TX_LOC_ FAULT_EN	PCS Tx Local Fault PCS Tx Local Fault 0b - PCS Tx Local Fault events do not cause an interrupt 1b - PCS Tx Local Fault events cause an interrupt
9 PCS_TX_Rem_ FAULT_EN	PCS Tx Remote Fault PCS Tx Remote Fault 0b - PCS Tx Remote Fault events do not cause an interrupt 1b - PCS Tx Remote Fault events cause an interrupt
10 PCS_RX_LOC_ FAULT_EN	PCS Rx Local Fault PCS Rx Local Fault 0b - PCS Rx Local Fault events do not cause an interrupt 1b - PCS Rx Local Fault events cause an interrupt
11 PCS_RX_Rem_ FAULT_EN	PCS Rx Remote Fault PCS Rx Remote Fault 0b - PCS Rx Remote Fault events do not cause an interrupt 1b - PCS Rx Remote Fault events cause an interrupt
12 PCS_HI_BER_E N	PCS High BER PCS High BER 0b - PCS High BER events do not cause an interrupt 1b - PCS High BER events cause an interrupt
13 PCS_BLK_LK_L OSS_EN	PCS Block Lock Loss PCS Block Lock Loss 0b - PCS Block Lock Loss events do not cause an interrupt 1b - PCS Block Lock Loss events cause an interrupt
14 TX_BUF_FAUL T_EN	Tx Buffer Fault Tx Buffer Fault 0b - PCS Tx Buffer Fault events do not cause an interrupt 1b - PCS Tx Buffer Fault events cause an interrupt
15 RX_BUF_FAUL T_EN	Rx Buffer Fault Rx Buffer Fault

## MDIO register spaces

Field	Function
	0b - PCS Rx Buffer Fault events do not cause an interrupt 1b - PCS Rx Buffer Fault events cause an interrupt

## 33.6.4.6 XFI Auto-Negotiation Interrupt Event (XFI\_VND\_AN\_INT)

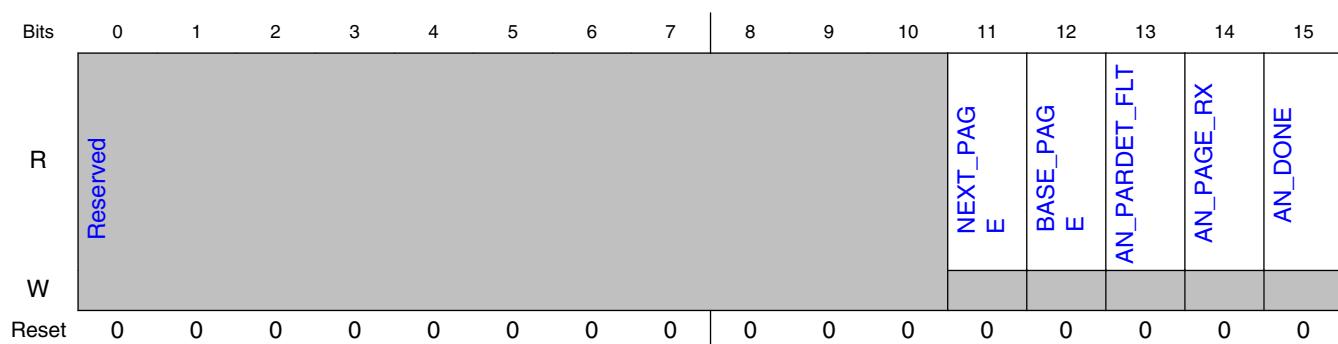
### 33.6.4.6.1 Offset

Register	Offset
XFI_VND_AN_INT	4h

### 33.6.4.6.2 Function

The XFI auto-negotiation interrupt event register contains interrupt events bits related to auto-negotiation. Note that XFI auto-negotiation interrupt event register bits are cleared when read.

### 33.6.4.6.3 Diagram



### 33.6.4.6.4 Fields

Field	Function
0-10	Reserved
—	
11 NEXT_PAGE	Next Page Received Next Page Received Asserted together with AN_PAGE_RX

*Table continues on the next page...*

Field	Function
	Cleared on register read 0b - No next page received 1b - Next page received and Partner Next Page Ability register is set
12 BASE_PAGE	Base Page Received Base Page Received Asserted together with AN_PAGE_RX Cleared on register read 0b - No base page received 1b - Base page received and Partner Ability register is set
13 AN_PARDET_F LT	Parallel detection fault Parallel detection fault Cleared on register read 0b - No fault detected 1b - Ambiguous link status detected while waiting on DME page receive
14 AN_PAGE_RX	Auto-negotiate page receive Auto-negotiate page receive Indicates the validity of the remote ability word (base page or next page) Cleared on register read 0b - An ability word was not received from the remote device 1b - An ability word was received from the remote device during backplane auto-negotiation
15 AN_DONE	Backplane auto-negotiation complete Backplane auto-negotiation complete Cleared on register read 0b - Backplane auto-negotiation not complete 1b - Backplane auto-negotiation complete

### 33.6.4.7 XFI Auto-Negotiation Interrupt Mask (XFI\_VND\_AN\_INT\_MSK)

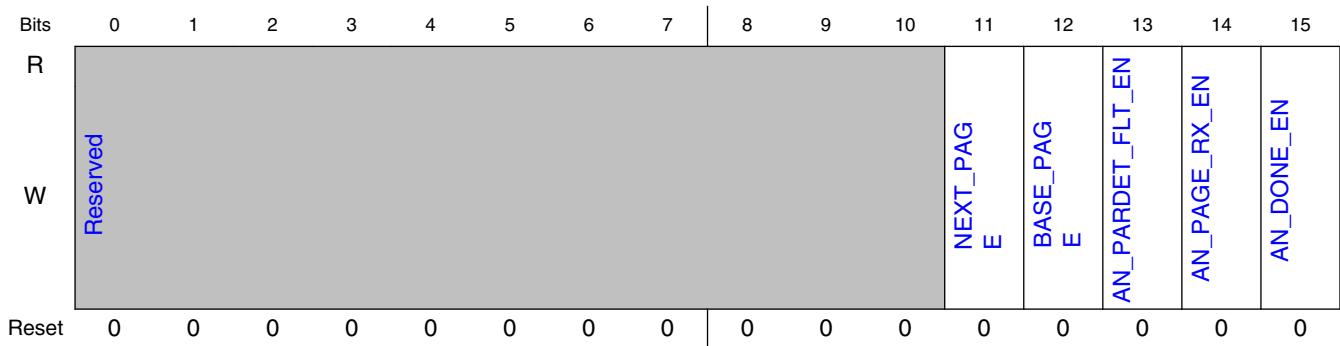
#### 33.6.4.7.1 Offset

Register	Offset
XFI_VND_AN_INT_MSK	5h

#### 33.6.4.7.2 Function

The XFI auto-negotiation interrupt mask register contains interrupt mask bits for the above auto-negotiate interrupt events.

### 33.6.4.7.3 Diagram



### 33.6.4.7.4 Fields

Field	Function
0-10 —	Reserved
11 NEXT_PAGE	Next Page Received Event Enable Next Page Received Event Enable 0b - AN Next Page Received will not cause an interrupt event 1b - AN Next Page Received will cause an interrupt event
12 BASE_PAGE	Base Page Received Event Enable Base Page Received Event Enable 0b - AN Base Page Received will not cause an interrupt event 1b - AN Base Page Received will cause an interrupt event
13 AN_PARDET_FLT_EN	AN Parallel Detection Fault Event Enable AN Parallel Detection Fault Event Enable 0b - AN Parallel Detection Fault will not cause an interrupt event 1b - AN Parallel Detection Fault will cause an interrupt event
14 AN_PAGE_RX_EN	AN Page Recieve Event Enable AN Page Recieve Event Enable 0b - AN Page Receive will not cause an interrupt event 1b - AN Page Receive will cause an interrupt event
15 AN_DONE_EN	AN Done Event Enable AN Done Event Enable 0b - AN done will not cause an interrupt event 1b - AN done will cause an interrupt event

### 33.6.4.8 XFI Link Training Interrupt Event (XFI\_VND\_LT\_INT)

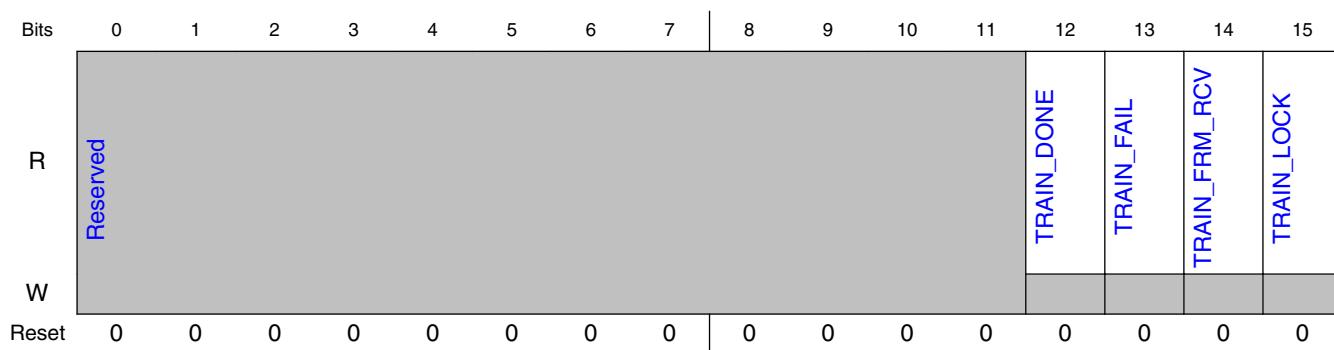
#### 33.6.4.8.1 Offset

Register	Offset
XFI_VND_LT_INT	6h

#### 33.6.4.8.2 Function

The XFI link training interrupt event register contains interrupt event bits related to link training. Note that XFI link training interrupt event register bits are cleared when read

#### 33.6.4.8.3 Diagram



#### 33.6.4.8.4 Fields

Field	Function
0-11	Reserved
—	
12 TRAIN_DONE	Link Training Completed Link Training Completed Cleared on register read 0b - Link training not complete 1b - Link training complete
13 TRAIN_FAIL	Link Taining Fault Indication Link Taining Fault Indication Cleared on register read 0b - Link training successful or not complete 1b - Link training failed

Table continues on the next page...

## MDIO register spaces

Field	Function
14 TRAIN_FRM_R CV	<p>Link Training Frame Receive Indication</p> <p>Link Training Frame Receive Indication</p> <p>Cleared on register read</p> <p>Note: during link training, training frames are received continuously, leading to assertion of this event every ~420 ns.</p> <p>0b - New coefficient update and status fields not available 1b - New coefficient update and status fields available</p>
15 TRAIN_LOCK	<p>Link Training Lock Indication</p> <p>Link Training Lock Indication</p> <p>Cleared on register read</p> <p>0b - Link training state machine is not locked 1b - Link training state machine is locked</p>

### 33.6.4.9 XFI Link Training Interrupt Mask (XFI\_VND\_LT\_INT\_MSK)

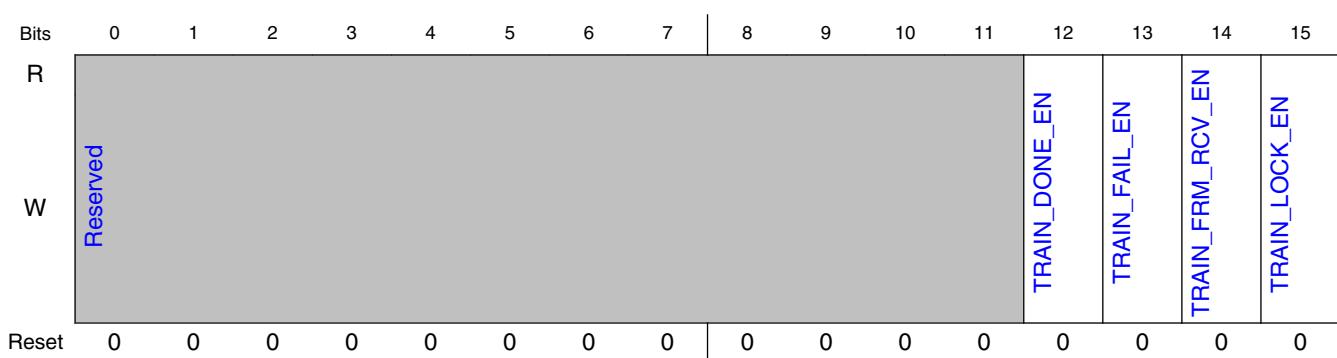
#### 33.6.4.9.1 Offset

Register	Offset
XFI_VND_LT_INT_MSK	7h

#### 33.6.4.9.2 Function

The XFI Link Training Interrupt Mask Register contains the mask bits for the above link training interrupt events.

#### 33.6.4.9.3 Diagram



### 33.6.4.9.4 Fields

Field	Function
0-11 —	Reserved
12 TRAIN_DONE_EN	Link Training Completed Interrupt Enable Link Training Completed Interrupt Enable 0b - Link Training Complete will not cause an interrupt event 1b - Link Training Complete will cause an interrupt event
13 TRAIN_FAIL_E_N	Link Training Fault Indication Interrupt Enable Link Training Fault Indication Interrupt Enable 0b - Link Training Fault Indication will not cause an interrupt event 1b - Link Training Fault Indication will cause an interrupt event
14 TRAIN_FRM_RCV_EN	Link Training Frame Receive Indication Interrupt Enable Link Training Frame Receive Indication Interrupt Enable 0b - Link Training Frame Receive Indication will not cause an interrupt event 1b - Link Training Frame Receive Indication will cause an interrupt event
15 TRAIN_LOCK_EN	Link Training Lock Indication Interrupt Enable Link Training Lock Indication Interrupt Enable 0b - Link Training Lock Indication will not cause an interrupt event 1b - Link Training Lock Indication will cause an interrupt event

## 33.6.5 MDIO\_KX\_PCS register descriptions

The 1000Base-KX PCS register space is selected when the associated SGMIIInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 03h.

This register space is also used for (vendor-specific) Clause 45 access to SGMII register aliases (starting at register address 8000h).

### 33.6.5.1 MDIO\_KX\_PCS memory map

MDIO\_KX\_PCS base address: 0h

## MDIO register spaces

Offset	Register	Width (In bits)	Access	Reset value
0h	KX PCS Control (KX_PCS_CR)	16	RW	1140h
1h	KX PCS Status (KX_PCS_SR)	16	RO	0009h
2h	KX PCS Device Identifier Upper (KX_PCS_DEV_ID)	16	RO	0083h
3h	KX PCS Device Identifier Lower (KX_PCS_DEV_ID_L)	16	RO	E400h
5h	KX PCS Devices In Package 0 (KX_PCS_DEV_PRES0)	16	RO	0000h
6h	KX PCS Devices In Package 1 (KX_PCS_DEV_PRES1)	16	RO	0088h
Eh	KX PCS Package Identifier Upper (KX_PCS_PKG_ID_U)	16	RO	0083h
Fh	KX PCS Package Identifier Lower (KX_PCS_PKG_ID_L)	16	RO	E400h
8000h	SGMII Control (C45_SGMII_CR)	16	RW	1140h
8001h	SGMII Status (C45_SGMII_SR)	16	RO	0009h
8002h	SGMII PHY Identifier Upper (C45_SGMII_PHY_ID_H)	16	RO	0083h
8003h	SGMII PHY Identifier Lower (C45_SGMII_PHY_ID_L)	16	RO	E400h
8004h	SGMII Device Ability for 1000Base-X (C45_SGMII_DEV_ABIL_1_KBX)	16	RW	01A0h
8004h	SGMII Device Ability for SGMII (C45_SGMII_DEV_ABIL_SGMII)	16	RW	01A0h
8005h	SGMII Partner Ability for 1000Base-X (C45_SGMII_LP_DEV_ABIL_1KBX)	16	RO	0000h
8005h	SGMII Partner Ability for SGMII (C45_SGMII_LP_DEV_ABIL_SGMII)	16	RO	0000h
8006h	SGMII AN Expansion (C45_SGMII_AN_EXP)	16	RO	0004h
8007h	SGMII Next Page Transmit (C45_SGMII_NP_TX)	16	RW	0000h
8008h	SGMII LP Next Page Receive (C45_SGMII_NP_RX)	16	RO	0000h
800Fh	SGMII Extended Status (C45_SGMII_XTND_STAT)	16	RU	0000h
8010h	SGMII Scratch (C45_SGMII_SCRATCH)	16	RW	0000h
8011h	SGMII Design Revision (C45_SGMII_REV)	16	RO	0001h
8012h	SGMII Link Timer Lower (C45_SGMII_LINK_TMR_L)	16	RW	12D0h
8013h	SGMII Link Timer Upper (C45_SGMII_LINK_TMR_H)	16	RW	0013h
8014h	SGMII IF Mode (C45_SGMII_IF_MODE)	16	RW	0000h

### 33.6.5.2 KX PCS Control (KX\_PCS\_CR)

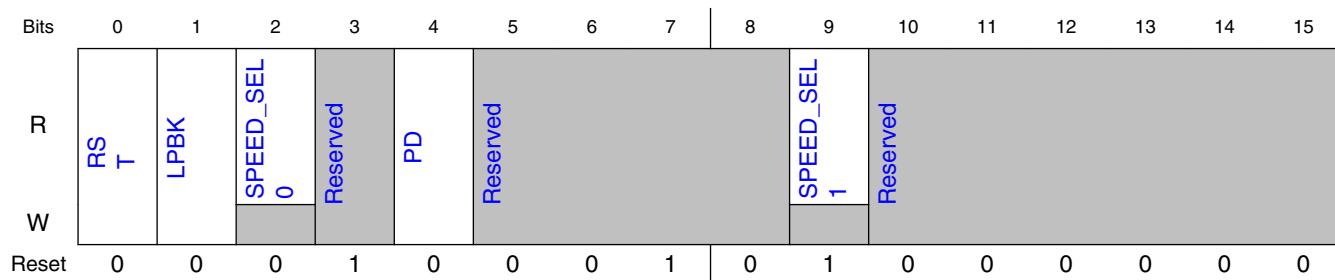
#### 33.6.5.2.1 Offset

Register	Offset
KX_PCS_CR	0h

### 33.6.5.2.2 Function

The 1000Base-KX PCS Control Register contains control bits used to enable/disable functions in the PCS, and to initiate commands such as reset.

### 33.6.5.2.3 Diagram



### 33.6.5.2.4 Fields

Field	Function
0 RST	Reset Self-Clearing Read / Write Reset Command Register. When set to 1, a synchronous reset pulse is generated which resets all the PCS state machines, the Comma detection function and the 8b/10b coder / decoder. Should be set to 0 (default) for normal operation.
1 LPBK	Loopback Loopback Command. 0b - Normal operation 1b - Reserved
2 SPEED_SEL0	Speed selection (LSB) MSB,LSB 1,0: 1000 Mb/s All others reserved Note: SGMII speed is controlled with the IF Mode register
3 —	Reserved
4 PD	Power down 0b - Normal operation 1b - Power down PCS. The PCS is held in internal soft reset until the bit is cleared again.
5-8 —	Reserved
9 SPEED_SEL1	Speed selection (MSB) See SPEED_SEL0
10-15 —	Reserved

### 33.6.5.3 KX PCS Status (KX\_PCS\_SR)

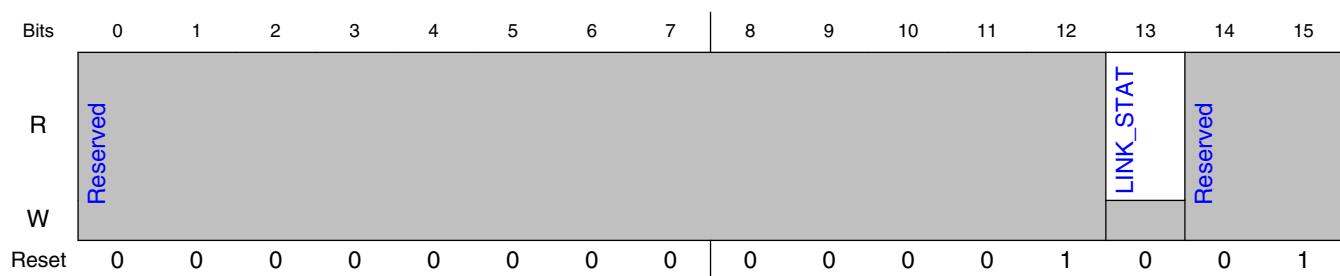
#### 33.6.5.3.1 Offset

Register	Offset
KX_PCS_SR	1h

#### 33.6.5.3.2 Function

The 1000Base-KX PCS Status Register contains status bits on the operation of the PCS.

#### 33.6.5.3.3 Diagram



#### 33.6.5.3.4 Fields

Field	Function
0-12 —	Reserved
13 LINK_STAT	Link status Link Status 0b - The link is not valid. If the link synchronization is lost a 0 is latched which is cleared only after a register read access 1b - This link is valid.
14-15 —	Reserved

### 33.6.5.4 KX PCS Device Identifier Upper (KX\_PCS\_DEV\_ID)

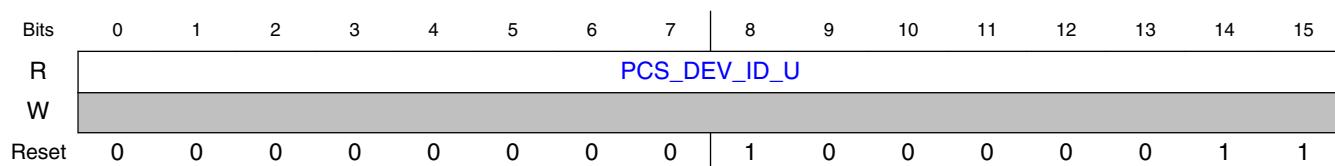
#### 33.6.5.4.1 Offset

Register	Offset
KX_PCS_DEV_ID	2h

#### 33.6.5.4.2 Function

The 1000Base-KX PCS Device Identifier Upper Register contains the upper half of the 32-bit Device Identifier.

#### 33.6.5.4.3 Diagram



#### 33.6.5.4.4 Fields

Field	Function
0-15	PCS Device Identifier Upper
PCS_DEV_ID_U	PCS Device Identifier Upper: OUI[3:18]

### 33.6.5.5 KX PCS Device Identifier Lower (KX\_PCS\_DEV\_ID\_L)

#### 33.6.5.5.1 Offset

Register	Offset
KX_PCS_DEV_ID_L	3h

### 33.6.5.5.2 Function

The 1000Base-KX PCS Identifier Lower Register contains the lower half of the 32-bit Device Identifier.

### 33.6.5.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									PCS_DEV_ID_L							
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.5.5.4 Fields

Field	Function
0-15 PCS_DEV_ID_L	PCS Device Identifier Lower PCS Device Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

## 33.6.5.6 KX PCS Devices In Package 0 (KX\_PCS\_DEV\_PRES0)

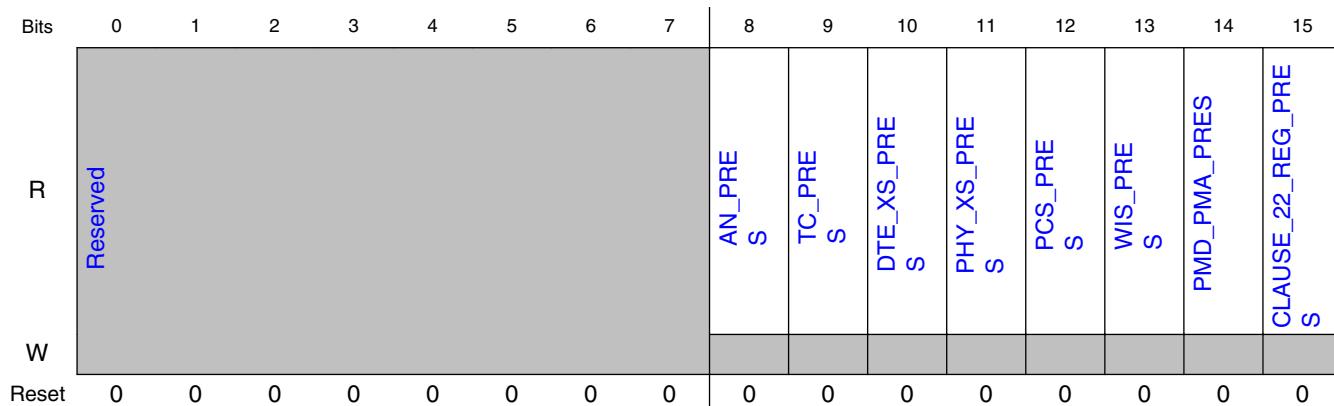
### 33.6.5.6.1 Offset

Register	Offset
KX_PCS_DEV_PRES0	5h

### 33.6.5.6.2 Function

The 1000Base-KX PCS Devices in Package 0 Register contains half of the PCS devices in package status.

### 33.6.5.6.3 Diagram



### 33.6.5.6.4 Fields

Field	Function
0-7 —	Reserved
8 AN_PRES	AN present 0b - Auto-Negotiation not present in package 1b - Auto-Negotiation present in package
9 TC_PRES	TC present 0b - TC not present in package 1b - TC present in package
10 DTE_XS_PRES	DTE XS present 0b - DTE XS not present in package 1b - DTE XS present in package
11 PHY_XS_PRES	PHY XS present 0b - PHY XS not present in package 1b - PHY XS present in package
12 PCS_PRES	PCS present 0b - PCS not present in package 1b - PCS present in package
13 WIS_PRES	WIS present 0b - WIS not present in package 1b - WIS present in package
14 PMD_PMA_PRES	PMD/PMA present 0b - PMA/PMD not present in package 1b - PMA/PMD present in package
15 CLAUSE_22_REG_PRES	Clause 22 registers present 0b - Clause 22 registers not present in package 1b - Clause 22 registers present in package

### 33.6.5.7 KX PCS Devices In Package 1 (KX\_PCS\_DEV\_PRES1)

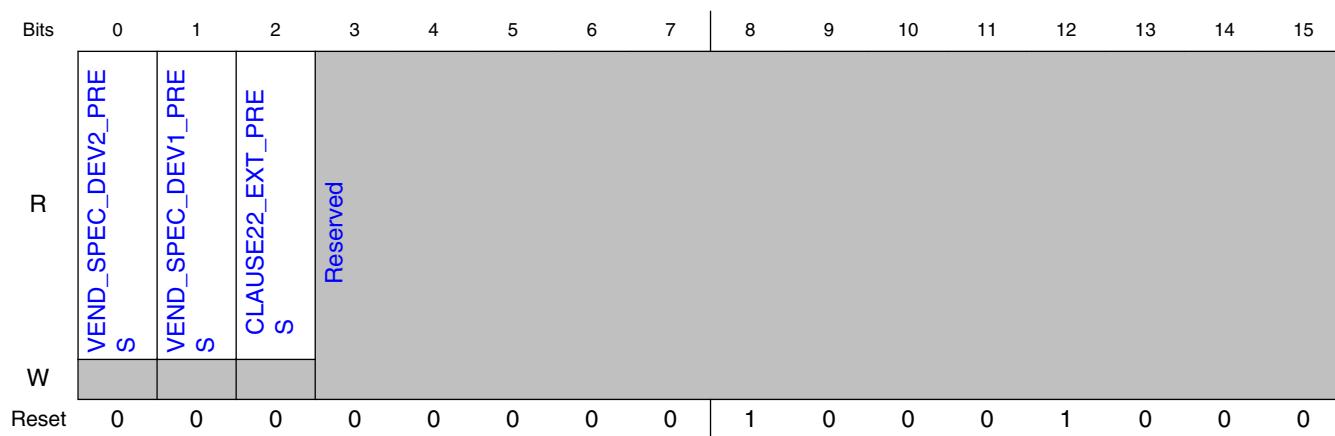
#### 33.6.5.7.1 Offset

Register	Offset
KX_PCS_DEV_PRES1	6h

#### 33.6.5.7.2 Function

The 1000Base-KX PCS Devices in Package 1 Register contains half of the PCS devices in package status.

#### 33.6.5.7.3 Diagram



#### 33.6.5.7.4 Fields

Field	Function
0 VEND_SPEC_DEV2_PRES	Vendor specific device 2 present 0b - Vendor specific device 2 not present in package 1b - Vendor specific device 2 present in package
1 VEND_SPEC_DEV1_PRES	Vendor specific device 1 present 0b - Vendor specific device 1 not present in package 1b - Vendor specific device 1 present in package
2 CLAUSE22_EXT_PRES	Clause 22 extension present 0b - Clause 22 extension not present in package 1b - Clause 22 extension present in package
3-15 —	Reserved

### 33.6.5.8 KX PCS Package Identifier Upper (KX\_PCS\_PKG\_ID\_U)

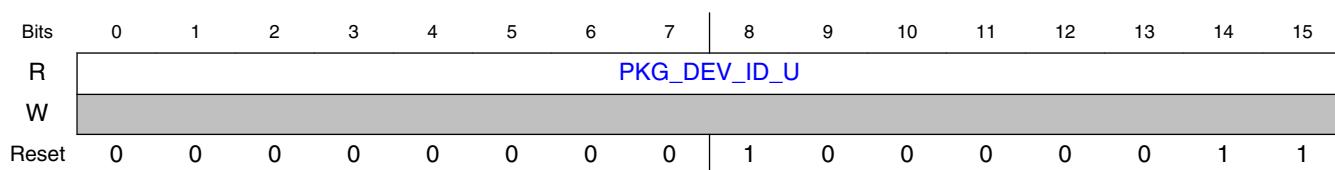
#### 33.6.5.8.1 Offset

Register	Offset
KX_PCS_PKG_ID_U	Eh

#### 33.6.5.8.2 Function

The 1000Base-KX PCS Package Device Identifier Upper Register contains the upper half of the 32-bit Package Identifier.

#### 33.6.5.8.3 Diagram



#### 33.6.5.8.4 Fields

Field	Function
0-15	Package Device Identifier Upper
PKG_DEV_ID_U	Package Device Identifier Upper: OUI[3:18]

### 33.6.5.9 KX PCS Package Identifier Lower (KX\_PCS\_PKG\_ID\_L)

#### 33.6.5.9.1 Offset

Register	Offset
KX_PCS_PKG_ID_L	Fh

### 33.6.5.9.2 Function

The KX PCS Package Identifier Lower Register contains the lower half of the 32-bit Package Identifier.

### 33.6.5.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	PKG_DEV_ID_L															
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.5.9.4 Fields

Field	Function
0-15 PKG_DEV_ID_L	Package Device Identifier Lower Package Device Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

## 33.6.5.10 SGMII Control (C45\_SGMII\_CR)

### 33.6.5.10.1 Offset

Register	Offset
C45_SGMII_CR	8000h

### 33.6.5.10.2 Function

The SGMII Control Register contains control bits used to enable/disable functions in the PCS, and to initiate commands such as reset.

### 33.6.5.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	RS T	LPBK	SPEED_SEL 0	AN_EN	PD	ISOLATE	RESTART_AN	FD	COL_TEST	SPEED_SEL 1	Reserved					
W	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0
Reset	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0

### 33.6.5.10.4 Fields

Field	Function
0 RST	Reset Self-Clearing Read / Write Reset Command Register. When set to 1, a synchronous reset pulse is generated which resets all the PCS state machines, the Comma detection function and the 8b/10b coder / decoder. Should be set to 0 (default) for normal operation.
1 LPBK	Loopback Loopback Command. 0b - Normal operation 1b - Reserved
2 SPEED_SEL0	Speed selection (LSB) Speed selection (LSB) All others reserved Note: SGMII speed is controlled with the IF Mode register MSB,LSB 1,0: 1000 Mb/s All others reserved
3 AN_EN	AN enable Auto-negotiation enable 0b - Disable auto-negotiation 1b - Enable auto-negotiation
4 PD	Power down Power down 0b - Normal operation 1b - Power down PCS. The PCS is held in internal soft reset until the bit is cleared again.
5 ISOLATE	Isolate Isolate 0b - Normal operation 1b - Reserved
6 RESTART_AN	Restart AN Restart auto-negotiation

Table continues on the next page...

## MDIO register spaces

Field	Function
	This bit is self-clearing 0b - Normal operation 1b - Restart auto-negotiation
7 FD	Full duplex Duplex mode. Read-only 0b - Reserved 1b - Full duplex
8 COL_TEST	Collision test Collision test. Read-only 0b - Disable COL signal test 1b - Reserved
9 SPEED_SEL1	Speed selection (MSB) Speed selection (MSB) See SPEED_SEL0
10-15 —	Reserved

### 33.6.5.11 SGMII Status (C45\_SGMII\_SR)

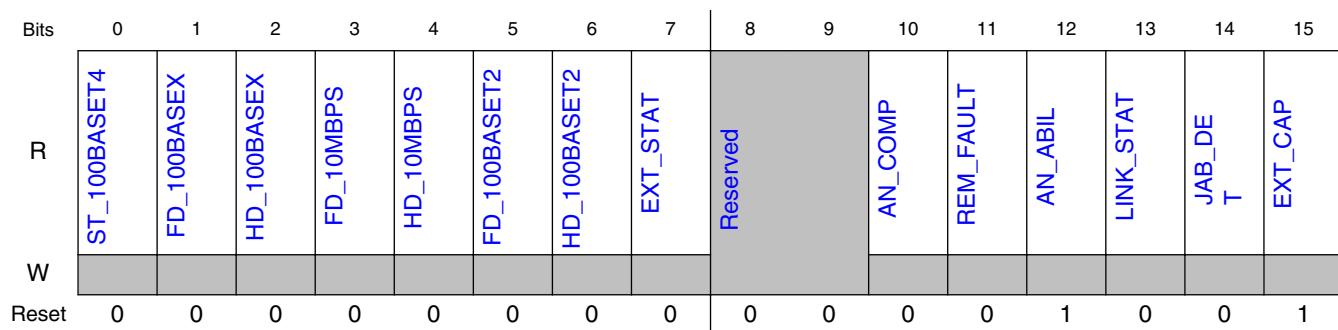
#### 33.6.5.11.1 Offset

Register	Offset
C45_SGMII_SR	8001h

#### 33.6.5.11.2 Function

The SGMII Status Register contains status bits on the operation of the PCS.

#### 33.6.5.11.3 Diagram



### 33.6.5.11.4 Fields

Field	Function
0	100Base-T4
ST_100Baset4	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T4 operation
1	100Base-X Full Duplex
FD_100BASEX	Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
2	100Base-X Half Duplex
HD_100BASEX	Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
3	10Mbps Full Duplex
FD_10MBPS	Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
4	10Mbps Half Duplex
HD_10MBPS	Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
5	100BaseT2 Full Duplex
FD_100Baset2	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
6	100BaseT2 Half Duplex
HD_100Baset2	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
7	Extended status
EXT_STAT	Read Only bit always set to 0 to indicate that the PCS does not implement an extended status register.
8-9	Reserved
—	
10	AN complete
AN_COMP	Auto-negotiation complete. Read Only Bit 0b - The Auto Negotiation process is not completed or Auto Negotiation is disabled 1b - The Auto Negotiation process is completed and that the Auto Negotiation control registers are valid.
11	Remote fault
REM_FAULT	Read Only Bit always set to 0. The PCS does not implement a PHY specific remote fault detection optional function.
12	AN ability
AN_ABIL	Auto Negotiation Ability. Read Only Bit set to „1. to indicate that the PCS supports Auto-Negotiation.
13	Link status
LINK_STAT	Link Status 0b - The link is not valid. If the link synchronization is lost a 0 is latched which is cleared only after a register read access 1b - This link is valid.
14	Jabber detection
JAB_DET	Read Only bit always set to 0, the Core does not support the optional Jabber detection function
15	Extended capability
EXT_CAP	Read Only bit set to 1. to indicate that the Core supports extended registers

### 33.6.5.12 SGMII PHY Identifier Upper (C45\_SGMII\_PHY\_ID\_H)

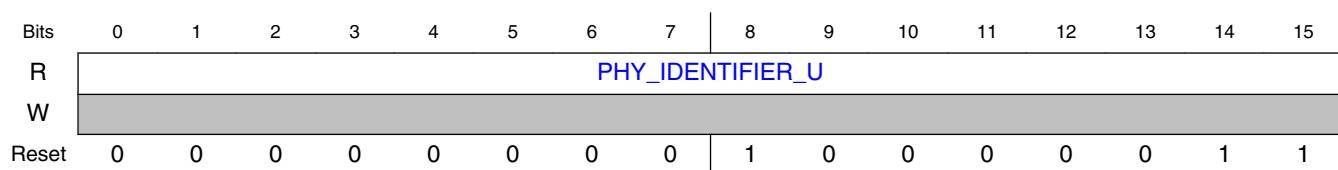
#### 33.6.5.12.1 Offset

Register	Offset
C45_SGMII_PHY_ID_H	8002h

#### 33.6.5.12.2 Function

The SGMII PHY Identifier Upper Register contains the upper half of the 32-bit PHY Identifier.

#### 33.6.5.12.3 Diagram



#### 33.6.5.12.4 Fields

Field	Function
0-15	PHY Identifier Upper
PHY_IDENTIFIER_U	PHY Identifier Upper: OUI[3:18]

### 33.6.5.13 SGMII PHY Identifier Lower (C45\_SGMII\_PHY\_ID\_L)

#### 33.6.5.13.1 Offset

Register	Offset
C45_SGMII_PHY_ID_L	8003h

### 33.6.5.13.2 Function

The SGMII PHY Identifier Lower Register contains the lower half of the 32-bit PHY Identifier.

### 33.6.5.13.3 Diagram

Bits	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R										PHY_IDENTIFIER_L							
W																	
Reset	1	1	1	0	0	1	0	0		0	0	0	0	0	0	0	

### 33.6.5.13.4 Fields

Field	Function
0-15 PHY_IDENTIFIER_L	PHY Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

## 33.6.5.14 SGMII Device Ability for 1000Base-X (C45\_SGMII\_DEV\_ABIL\_1KBX)

### 33.6.5.14.1 Offset

Register	Offset
C45_SGMII_DEV_ABIL_1KBX	8004h

### 33.6.5.14.2 Function

The SGMII Device Ability Register contains the control bits used to advertise the device abilities to the Link Partner during auto-negotiation for 1000Base-X mode.

### 33.6.5.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	NP	ACK	RF2	RF1	Reserved			PS2	PS1	HD	FD	Reserved				
W	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0
Reset																

### 33.6.5.14.4 Fields

Field	Function
0 NP	Next page Next page support. Set to 1 to indicate next page can be transferred following base page exchange
1 ACK	Ack Read only bit set to 1 when device has received three consecutive matching ability values from the link partner.
2 RF2	Remote fault 2 Fault condition advertised by the device: RF1=0 / RF2=0: No error, link is OK (Reset condition). RF1=0 / RF2=1: Device advertises that it is Off Line. RF1=1 / RF2=0: Device advertises a link failure condition.. RF1=1 / RF2=1: Device advertises an Auto Negotiation error. Note: the PCS does not interpret or set these bits. It is up to the application to implement the fault functions as needed.
3 RF1	Remote Fault 1 See RF2
4-6 —	Reserved
7 PS2	Pause 2 Pause bits both set to 1 (Reset Value) to advertise that the Core supports pause on both transmit and receive. Can be set to any other value to advertise other flow control capabilities.
8 PS1	Pause 1 See PS2
9 HD	Half Duplex Half Duplex Enable. Read only bit set to 1 when the device advertises that it supports Half Duplex Mode of operation.
10 FD	Full Duplex Full Duplex Enable. Set to 1 when the device advertises that it supports Full Duplex Mode of operation
11-15 —	Reserved

### 33.6.5.15 SGMII Device Ability for SGMII (C45\_SGMII\_DEV\_ABIL\_SGMII)

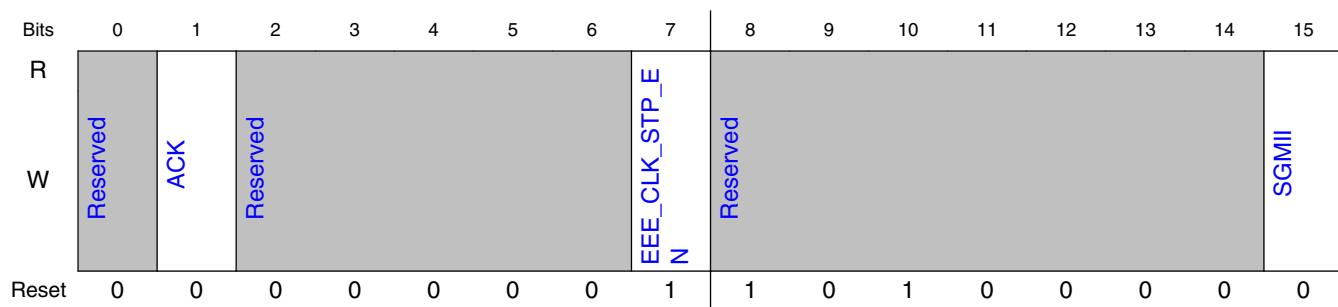
#### 33.6.5.15.1 Offset

Register	Offset
C45_SGMII_DEV_ABIL_SGMII	8004h

#### 33.6.5.15.2 Function

The SGMII Device Ability Register contains the control bits used to advertise the device abilities to the Link Partner during auto-negotiation for SGMII mode.

#### 33.6.5.15.3 Diagram



#### 33.6.5.15.4 Fields

Field	Function
0	Reserved
—	
1	Ack
ACK	Read only bit set to 1 when device has received three consecutive matching ability values from the link partner
2-6	Reserved. Should be set to 0
—	
7	EEE Clock Stop Enable
EEE_CLK_STP_EN	EEE Clock Stop Enable

Table continues on the next page...

## MDIO register spaces

Field	Function
	Should be set to 0 before SGMII Auto-Negotiation enable/restart, as this device does not support EEE clock stop. 0b - EEE Clock Stop Disabled 1b - EEE Clock Stop Enabled
8-14 —	Reserved. Should be set to 0
15 SGMII	SGMII mode. Should be set to 1.

### 33.6.5.16 SGMII Partner Ability for 1000Base-X (C45\_SGMII\_LP\_DEV\_ABIL\_1KBX)

#### 33.6.5.16.1 Offset

Register	Offset
C45_SGMII_LP_DEV_ABIL_1KBX	8005h

#### 33.6.5.16.2 Function

The SGMII Partner Ability Register contains the bits advertised by the Link Partner during auto-negotiation for 1000Base-X mode.

#### 33.6.5.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	NP	ACK	RF2	RF1	Reserved			PS2	PS1	HD	FD	Reserved				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 33.6.5.16.4 Fields

Field	Function
0 NP	Next Page Next page support.
1 Ack	Ack

*Table continues on the next page...*

Field	Function
ACK	Set to 1 when link partner has received three consecutive matching ability values from the device.
2	Remote Fault 2
RF2	Fault condition advertised by the link partner: RF1/RF2: 00: No error, link is OK (Reset condition). 01: Off Line. 10: Link failure 11: Auto Negotiation error.
3	Remote Fault 1
RF1	See RF2
4-6	Reserved
—	
7	Pause 2
PS2	Pause capability of link partner (ASM_DIR:PAUSE) 0b00: No Pause 0b11: Symmetric Pause 0b10: Asymmetric Pause toward link partner 0b11: Both Symmetric Pause and Asymmetric Pause toward local device
8	Pause 1
PS1	See PS2
9	Half Duplex
HD	Half Duplex support
10	Full Duplex
FD	Full Duplex support
11-15	Reserved
—	

### 33.6.5.17 SGMII Partner Ability for SGMII (C45\_SGMII\_LP\_DEV\_ABIL\_SGMII)

#### 33.6.5.17.1 Offset

Register	Offset
C45_SGMII_LP_DEV_ABIL_SGMII	8005h

### 33.6.5.17.2 Function

The SGMII Partner Ability Register contains the capability status of the SGMII link partner.

### 33.6.5.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	COP_LNK_STAT	ACK	Reserved	COP_DUPL	COP_SPD		EEE_CAP									
W									Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.5.17.4 Fields

Field	Function
0 COP_LNK_STA T	Copper Link Status Read only bit, used by the SGMII PHY to advertise the Link Partner Copper status: 0b - Copper interface link is down 1b - Copper interface link is up
1 ACK	Ack Read only bit set to „1. when the Link Partner Copper Interface advertises that it has received three consecutive matching ability values from the device
2 —	Always 0
3 COP_DUPL	Copper Duplex Read only bit, used by the SGMII PHY to advertise the Link Partner Copper duplex capability: 0b - Copper Interface resolved to Half-Duplex 1b - Copper Interface resolved to Full-Duplex
4-5 COP_SPD	Copper speed Read only bits, used by the SGMII PHY to advertise the Link Partner Copper interface speed 00b - Copper Interface Speed is 10Mbps 01b - Copper Interface Speed is 100Mbps 10b - Copper Interface Speed is Gigabit 11b - Reserved
6 EEE_CAP	EEE Capability

Table continues on the next page...

Field	Function
	0b - EEE not supported 1b - EEE supported
7 EEE_CLK_STO P_CAP	EEE Clock Stop Capability EEE Clock Stop Capability 0b - EEE Clock Stop not supported 1b - EEE Clock Stop supported
8-15 —	Reserved

### 33.6.5.18 SGMII AN Expansion (C45\_SGMII\_AN\_EXP)

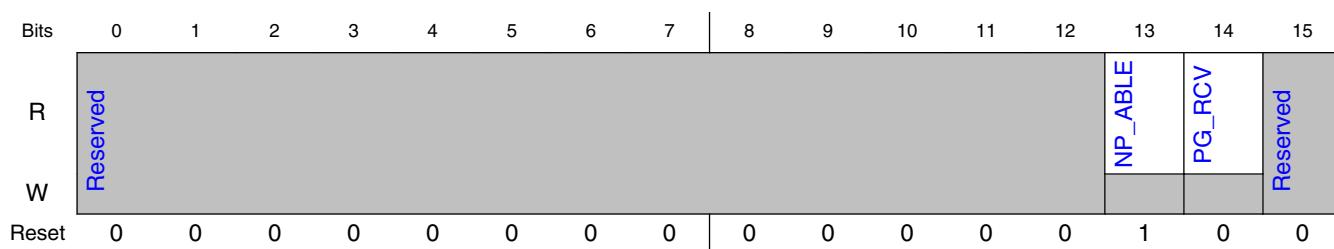
#### 33.6.5.18.1 Offset

Register	Offset
C45_SGMII_AN_EXP	8006h

#### 33.6.5.18.2 Function

The SGMII AN Expansion Register contains status bits indicating the PCS next page auto-negotiation capability and status.

#### 33.6.5.18.3 Diagram



#### 33.6.5.18.4 Fields

Field	Function
0-12 —	Reserved

Table continues on the next page...

## MDIO register spaces

Field	Function
13 NP_ABLE	Next Page Able Read Only bit set to 1 to indicate the PCS does support the Next Page function
14 PG_RCV	Page Received Set to 1 to indicate that a new page has been received with new partner ability available in the PCS register PARTNER_ABILITY. The bit is set to 0 (Reset value) when the system management agent performs a read access.
15 —	Reserved

### 33.6.5.19 SGMII Next Page Transmit (C45\_SGMII\_NP\_TX)

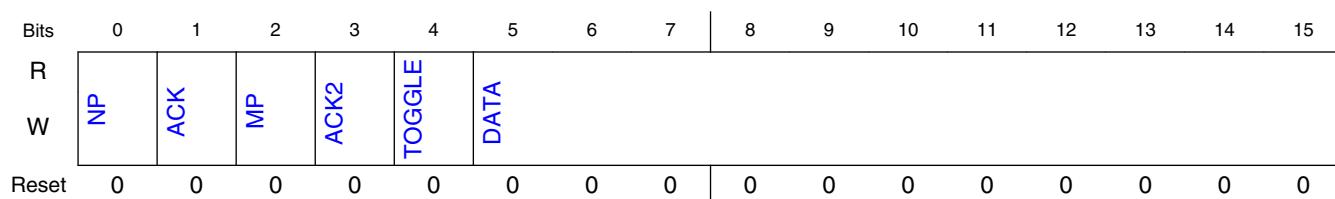
#### 33.6.5.19.1 Offset

Register	Offset
C45_SGMII_NP_TX	8007h

#### 33.6.5.19.2 Function

The SGMII NP TX Register contains next page data to transfer to the remote devide. Writing to this register initiates a next page exchange (sets mr\_np\_loaded variable).

#### 33.6.5.19.3 Diagram



#### 33.6.5.19.4 Fields

Field	Function
0 NP	Next page Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1	Ack

*Table continues on the next page...*

Field	Function
ACK	Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2	Message page
MP	Message page (1) or unformatted page (0) format.
3	Ack 2
ACK2	Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4	Toggle
TOGGLE	The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15	Data
DATA	11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

### 33.6.5.20 SGMII LP Next Page Receive (C45\_SGMII\_NP\_RX)

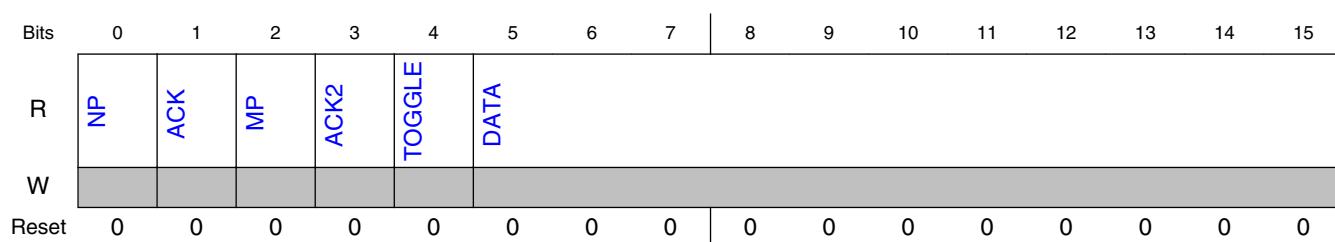
#### 33.6.5.20.1 Offset

Register	Offset
C45_SGMII_NP_RX	8008h

#### 33.6.5.20.2 Function

The SGMII NP RX Register contains the next page data received from the remote device during the latest next page exchange. The value is overridden with every new next page. Next page transfers are controlled by the application.

#### 33.6.5.20.3 Diagram



### 33.6.5.20.4 Fields

Field	Function
0	Next page
NP	Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1	Ack
ACK	Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2	Message page
MP	Message page (1) or unformatted page (0) format.
3	Ack 2
ACK2	Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4	Toggle
TOGGLE	The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15	Data
DATA	11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

### 33.6.5.21 SGMII Extended Status (C45\_SGMII\_XTND\_STAT)

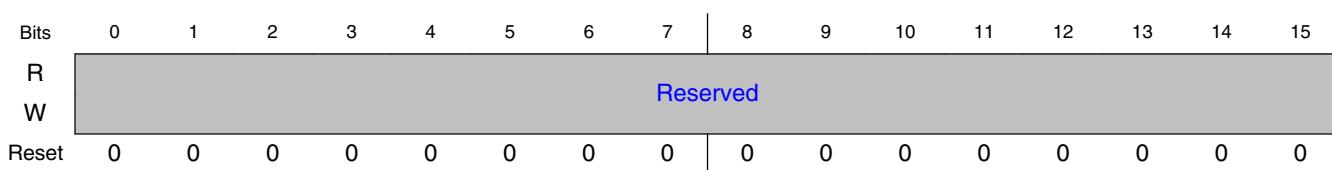
#### 33.6.5.21.1 Offset

Register	Offset
C45_SGMII_XTND_SSTAT	800Fh

#### 33.6.5.21.2 Function

The SGMII Extended Status Register is reserved, as this device does not implement the optional extended status registers.

#### 33.6.5.21.3 Diagram



### 33.6.5.21.4 Fields

Field	Function
0-15	Reserved. Always 0
—	

### 33.6.5.22 SGMII Scratch (C45\_SGMII\_SCRATCH)

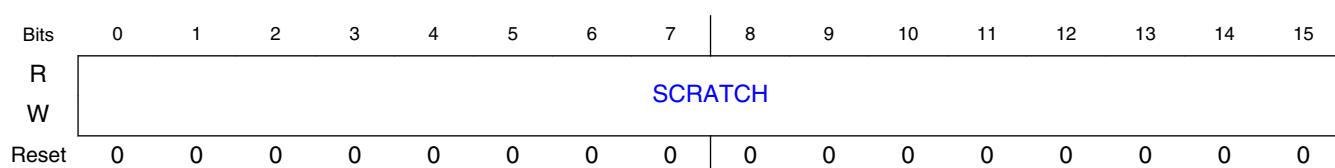
#### 33.6.5.22.1 Offset

Register	Offset
C45_SGMII_SCRATCH	8010h

#### 33.6.5.22.2 Function

The SGMII Scratch Register provides a memory location available to test register read and write operations.

#### 33.6.5.22.3 Diagram



#### 33.6.5.22.4 Fields

Field	Function
0-15	Scratch
SCRATCH	Scratch field.

### 33.6.5.23 SGMII Design Revision (C45\_SGMII\_REV)

### 33.6.5.23.1 Offset

Register	Offset
C45_SGMII_REV	8011h

### 33.6.5.23.2 Function

The SGMII Revision Register contains revision information for the PCS.

### 33.6.5.23.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

### 33.6.5.23.4 Fields

Field	Function
0-7	Reserved
—	
8-11	Major revision
IP_MJ	Major revision
12-15	Minor revision
IP_MN	Minor revision

## 33.6.5.24 SGMII Link Timer Lower (C45\_SGMII\_LINK\_TMR\_L)

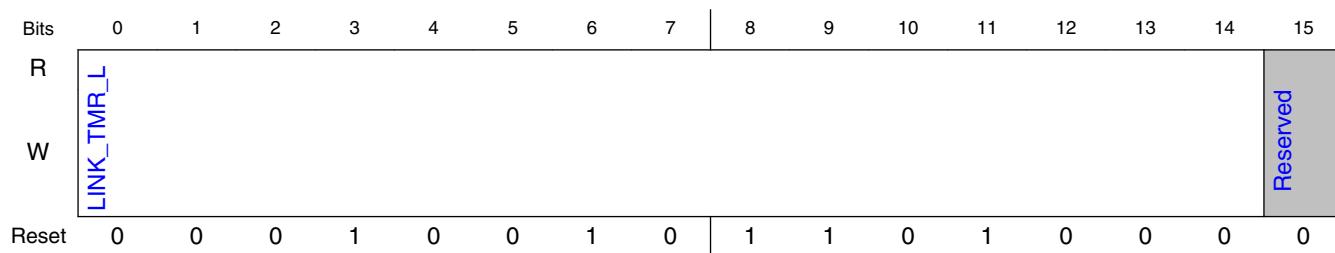
### 33.6.5.24.1 Offset

Register	Offset
C45_SGMII_LINK_TMR_L	8012h

### 33.6.5.24.2 Function

The SGMII Link Timer Register Lower contains bits 15:1 of the link timer value.

### 33.6.5.24.3 Diagram



### 33.6.5.24.4 Fields

Field	Function
0-14 LINK_TMR_L	Link Timer Lower Link timer[15:1]
15 —	Reserved

## 33.6.5.25 SGMII Link Timer Upper (C45\_SGMII\_LINK\_TMR\_H)

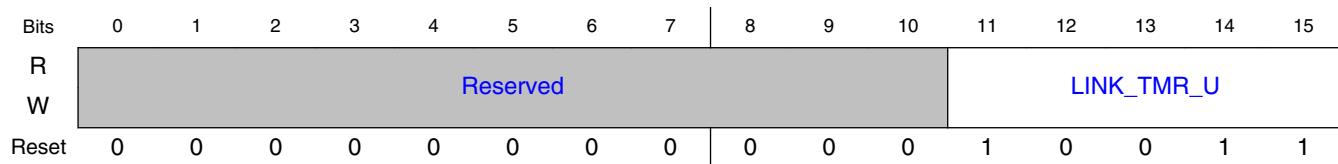
### 33.6.5.25.1 Offset

Register	Offset
C45_SGMII_LINK_TMR_H	8013h

### 33.6.5.25.2 Function

The SGMII Link Timer Register Upper contains bits 20:16 of the link timer value.

### 33.6.5.25.3 Diagram



### 33.6.5.25.4 Fields

Field	Function
0-10 —	Reserved
11-15	Link Timer Upper
LINK_TMR_U	Link timer[20:16]

## 33.6.5.26 SGMII IF Mode (C45\_SGMII\_IF\_MODE)

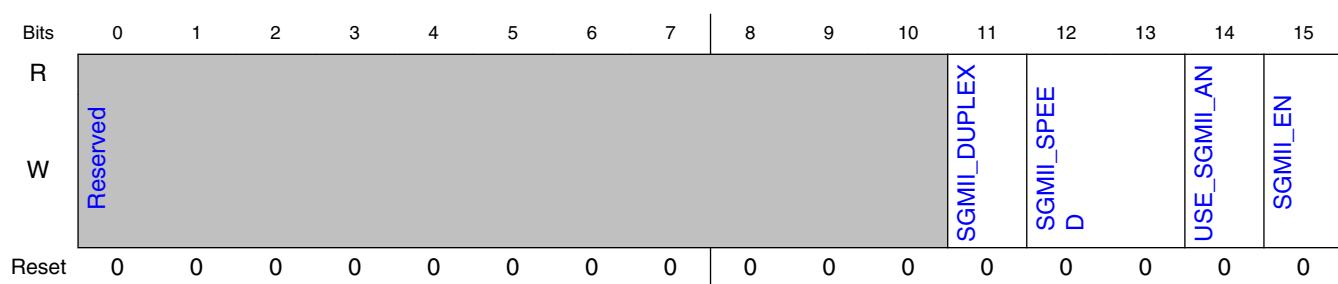
### 33.6.5.26.1 Offset

Register	Offset
C45_SGMII_IF_MODE	8014h

### 33.6.5.26.2 Function

The SGMII IF Mode Register contains control bits to set the interface mode.

### 33.6.5.26.3 Diagram



### 33.6.5.26.4 Fields

Field	Function
0-10 —	Reserved
11 SGMII_DUPLEX	SGMII Duplex SGMII Duplex Mode: Bit ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 0b - Full duplex enabled (default)
12-13 SGMII_SPEED	SGMII Speed SGMII Speed. When the PCS operates in SGMII mode (SGMII_EN set to 1) and is programmed not to be automatically configured (USE_SGMII_AN set to 0), sets the PCS speed of operation: Bits ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 00b - 10Mbps 01b - 100Mbps 10b - Gigabit (1G or 2.5G) 11b - Reserved
14 USE_SGMII_AN	Use SGMII AN Use the SGMII Auto-Negotiation Results to Program the PCS Speed. When set to 0 (Reset Value), the PCS operation should be programmed with the register bit SGMII_SPEED and SGMII_DUPLEX. When set to 1, the PCS operation is automatically programmed with the Partner abilities advertised during Auto-Negotiation. Ignored when SGMII_EN is set to 0.
15 SGMII_EN	SGMII Enable SGMII Mode Enable. When set to '0' (Reset Value), the PCS operates in standard 1000Base-X Gigabit mode, when set to '1', the PCS operates in SGMII Mode

### 33.6.6 MDIO\_KX\_AN register descriptions

The 1000Base-KX auto-negotiation register space is selected when the associated SGMIIInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 07h.

#### 33.6.6.1 MDIO\_KX\_AN memory map

MDIO\_KX\_AN base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	KX AN Control (KX_AN_CR)	16	RW	0000h
1h	KX AN Status (KX_AN_SR)	16	RO	0040h

*Table continues on the next page...*

## MDIO register spaces

Offset	Register	Width (In bits)	Access	Reset value
2h	KX AN Device Identifier Upper (KX_AN_DEV_ID_H)	16	RO	0083h
3h	KX AN Device Identifier Lower (KX_AN_DEV_ID_L)	16	RO	E400h
5h	KX AN Devices in Package 0 (KX_AN_DEV_PRES0)	16	RO	0088h
6h	KX AN Devices in Package 1 (KX_AN_DEV_PRES1)	16	RO	0000h
Eh	KX AN Package Identifier Upper (KX_AN_PKG_ID_H)	16	RO	0083h
Fh	KX AN Package Identifier Lower (KX_AN_PKG_ID_L)	16	RO	E400h
10h	KX AN Advertisement 0 (KX_AN_ADVERT0)	16	RW	0C01h
11h	KX AN Advertisement 1 (KX_AN_ADVERT1)	16	RW	003Fh
12h	KX AN Advertisement 2 (KX_AN_ADVERT2)	16	RW	0000h
13h	KX AN LP Base Page Ability 0 (KX_AN_LP_BASE_PG_ABIL0)	16	RO	0000h
14h	KX AN LP Base Page Ability 1 (KX_AN_LP_BASE_PG_ABIL1)	16	RO	0000h
15h	KX AN LP Base Page Ability 2 (KX_AN_LP_BASE_PG_ABIL2)	16	RO	0000h
16h	KX AN XNP Transmit 0 (KX_AN_XNP_TX0)	16	RO	2001h
17h	KX AN XNP Transmit 1 (KX_AN_XNP_TX1)	16	RO	0000h
18h	KX AN XNP Transmit 2 (KX_AN_XNP_TX2)	16	RO	0000h
19h	KX AN LP XNP Ability 0 (KX_AN_LP_XNP_ABIL0)	16	RO	0000h
1Ah	KX AN LP XNP Ability 1 (KX_AN_LP_XNP_ABIL1)	16	RO	0000h
1Bh	KX AN LP XNP Ability 2 (KX_AN_LP_XNP_ABIL2)	16	RO	0000h
30h	KX Backplane Ethernet Status (KX_BP_STAT)	16	RO	0001h
8000h	KX Millisecond Count (KX_MS_CNT)	16	RW	9896h

### 33.6.6.2 KX AN Control (KX\_AN\_CR)

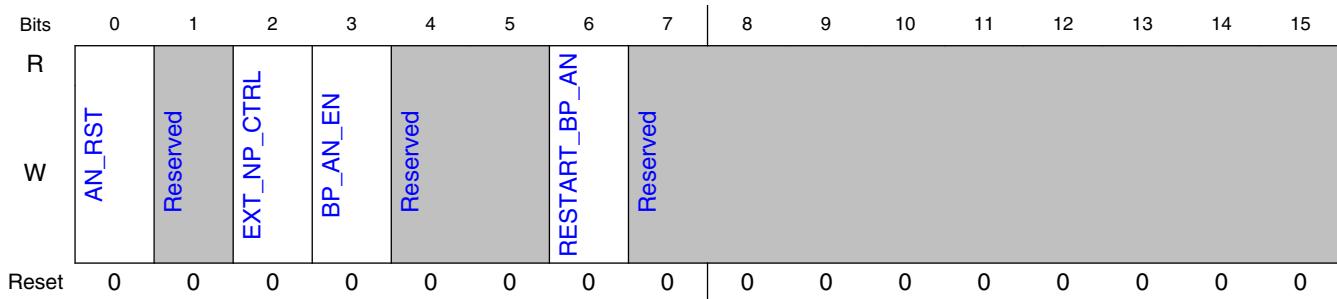
#### 33.6.6.2.1 Offset

Register	Offset
KX_AN_CR	0h

#### 33.6.6.2.2 Function

The 1000Base-KX AN Control Register contains control bits used to enable/disable functions in the PCS, and to initiate commands such as reset.

### 33.6.6.2.3 Diagram



### 33.6.6.2.4 Fields

Field	Function
0 AN_RST	AN Reset AN reset This bit is self-clearing 0b - AN normal operation 1b - AN reset
1 —	Reserved
2 EXT_NP_CTRL	Extended Next Page Control Extended Next Page Control When enabled (1) transmission of next page with non-null code field is possible. The next page registers should be initialized and must be set (handshaking) every time a next page is received. When disabled (0) only null next page is transmitted in response to received next pages from link partner. Note: the next page data registers (AN XNP) are writeable only if this bit is set. 0b - Extended next pages are disabled 1b - Extended next pages are enabled
3 BP_AN_EN	Backplane AN Enable Backplane auto-negotiation enable 0b - Disable auto-negotiation 1b - Enable auto-negotiation
4-5 —	Reserved
6 RESTART_BP_AN	Restart backplane AN Restart backplane auto-negotiation This bit is self-clearing 0b - Auto-Negotiation in process, disabled, or not supported 1b - Restart Auto-Negotiation process
7-15 —	Reserved

### 33.6.6.3 KX AN Status (KX\_AN\_SR)

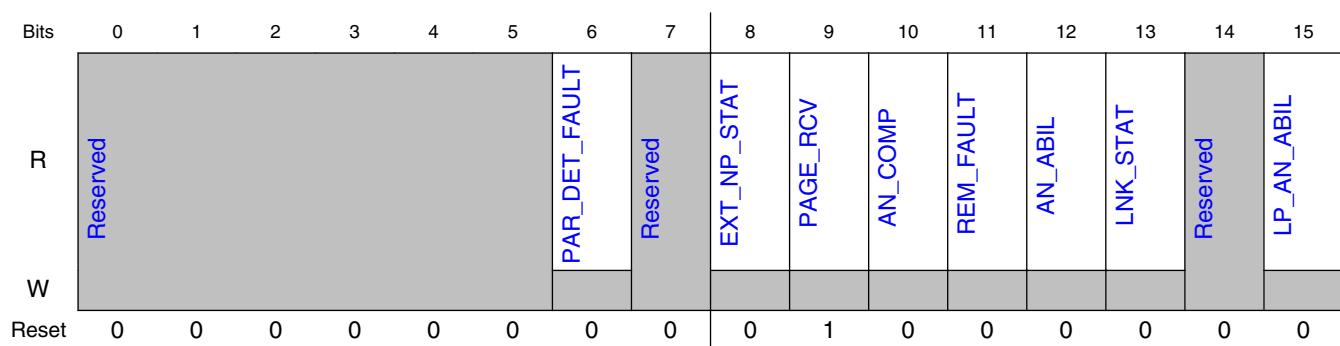
#### 33.6.6.3.1 Offset

Register	Offset
KX_AN_SR	1h

#### 33.6.6.3.2 Function

The 1000Base-KX AN Status Register contains status bits for the 1000Base-KX auto-negotiate function.

#### 33.6.6.3.3 Diagram



#### 33.6.6.3.4 Fields

Field	Function
0-5	Reserved
—	
6 PAR_DET_FAU LT	Parallel Detection Fault Parallel Detection Fault Once set, stays set until the register is read. 0b - A fault has not been detected via the parallel detection function. 1b - A fault has been detected via the parallel detection function.
7	Reserved
—	
8	Extended Next Page Status

*Table continues on the next page...*

Field	Function
EXT_NP_STAT	<p>Extended Next Page Status</p> <p>This bit is a copy of AN Control[EXT_NP_CTRL]</p> <p>0b - Extended next pages disabled 1b - Extended next pages enabled</p>
9 PAGE_RCV	<p>Page received</p> <p>Page received</p> <p>Once set, stays set until the register is read</p> <p>0b - Page has not been received 1b - Page has been received</p>
10 AN_COMP	<p>AN Complete</p> <p>Auto-negotiation complete</p> <p>0b - Auto-negotiation process not completed 1b - Auto-negotiation process completed</p>
11 REM_FAULT	<p>Remote Fault</p> <p>Remote Fault</p> <p>Once set, stays set until the register is read</p> <p>0b - No remote fault condition detected 1b - Remote fault condition detected</p>
12 AN_ABIL	<p>AN Ability</p> <p>Auto-negotiation Ability</p> <p>Always set to 1 to indicate that PCS is able to perform auto-negotiation</p>
13 LNK_STAT	<p>Link Status</p> <p>Link Status</p> <p>Once cleared, stays cleared until the register is read</p> <p>0b - Link is down 1b - Link is up</p>
14 —	Reserved
15 LP_AN_ABIL	<p>Link Partner AN Ability</p> <p>Link Partner Auto-Negotiation Ability</p> <p>0b - Link Partner is not able to perform auto-negotiation, or Link Partner ability not yet captured 1b - Link Partner is able to perform auto-negotiation</p>

### 33.6.6.4 KX AN Device Identifier Upper (KX\_AN\_DEV\_ID\_H)

#### 33.6.6.4.1 Offset

Register	Offset
KX_AN_DEV_ID_H	2h

### 33.6.6.4.2 Function

The 1000Base-KX AN Device Identifier Upper Register contains the upper half of the 32-bit device identifier.

### 33.6.6.4.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									AN_DEV_ID_U							
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1

### 33.6.6.4.4 Fields

Field	Function
0-15	AN Device Identifier Upper
AN_DEV_ID_U	PCS Device Identifier Upper: OUI[3:18]

## 33.6.6.5 KX AN Device Identifier Lower (KX\_AN\_DEV\_ID\_L)

### 33.6.6.5.1 Offset

Register	Offset
KX_AN_DEV_ID_L	3h

### 33.6.6.5.2 Function

The 1000Base-KX AN Device Identifier Lower Register contains the lower half of the 32-bit Device Identifier.

### 33.6.6.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	AN_DEV_ID_L															
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.6.5.4 Fields

Field	Function
0-15 AN_DEV_ID_L	AN Device ID Lower PCS Device Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

## 33.6.6.6 KX AN Devices in Package 0 (KX\_AN\_DEV\_PRES0)

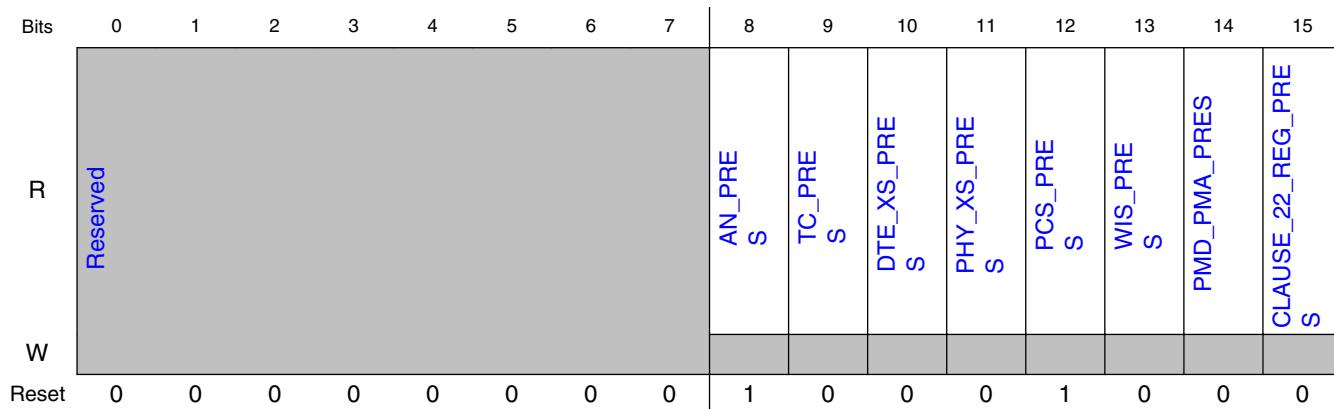
### 33.6.6.6.1 Offset

Register	Offset
KX_AN_DEV_PRES0	5h

### 33.6.6.6.2 Function

The 1000Base-KX AN Devices in Package 0 Register contains half of the AN devices in package status.

### 33.6.6.3 Diagram



### 33.6.6.4 Fields

Field	Function
0-7 —	Reserved
8 AN_PRES	AN Present 0b - Auto-Negotiation not present in package 1b - Auto-Negotiation present in package
9 TC_PRES	TC Present 0b - TC not present in package 1b - TC present in package
10 DTE_XS_PRES	DTE XS Present 0b - DTE XS not present in package 1b - DTE XS present in package
11 PHY_XS_PRES	PHY XS Present 0b - PHY XS not present in package 1b - PHY XS present in package
12 PCS_PRES	PCS Present 0b - PCS not present in package 1b - PCS present in package
13 WIS_PRES	WIS Present 0b - WIS not present in package 1b - WIS present in package
14 PMD_PMA_PRES	PMD/PMA Present 0b - PMA/PMD not present in package 1b - PMA/PMD present in package
15 CLAUSE_22_REG_PRES	Clause 22 register present 0b - Clause 22 registers not present in package 1b - Clause 22 registers present in package

### 33.6.6.7 KX AN Devices in Package 1 (KX\_AN\_DEV\_PRES1)

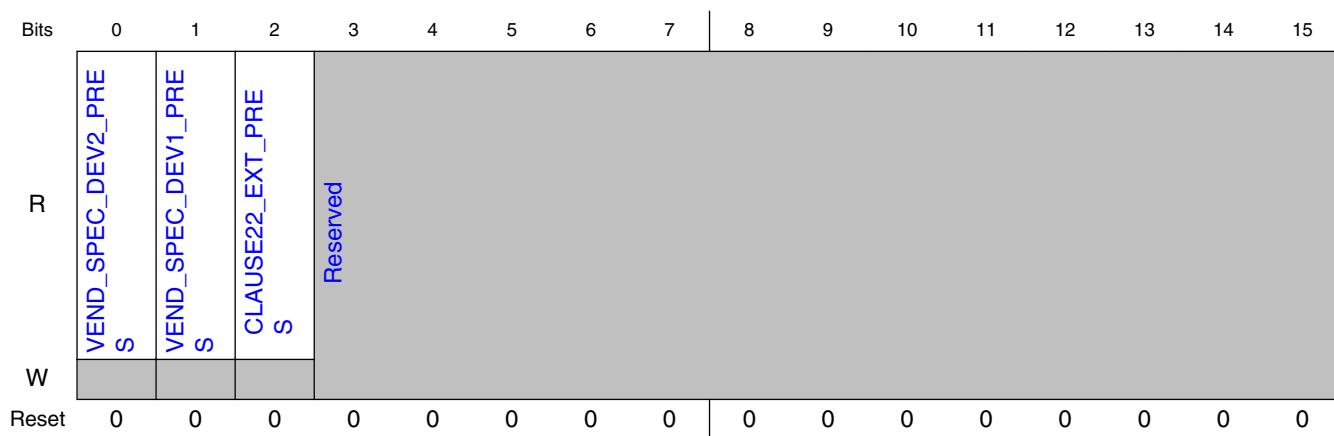
#### 33.6.6.7.1 Offset

Register	Offset
KX_AN_DEV_PRES1	6h

#### 33.6.6.7.2 Function

The 1000Base-KX AN Devices in Package 1 Register contains half of the AN devices in package status.

#### 33.6.6.7.3 Diagram



#### 33.6.6.7.4 Fields

Field	Function
0 VEND_SPEC_DEV2_PRES	Vendor Specific Device 2 Present 0b - Vendor specific device 2 not present in package 1b - Vendor specific device 2 present in package
1 VEND_SPEC_DEV1_PRES	Vendor Specific Device 1 Present 0b - Vendor specific device 1 not present in package 1b - Vendor specific device 1 present in package
2 CLAUSE22_EXTENSION_PRES	Clause 22 Extension Present 0b - Clause 22 extension not present in package 1b - Clause 22 extension present in package
3-15 —	Reserved

### 33.6.6.8 KX AN Package Identifier Upper (KX\_AN\_PKG\_ID\_H)

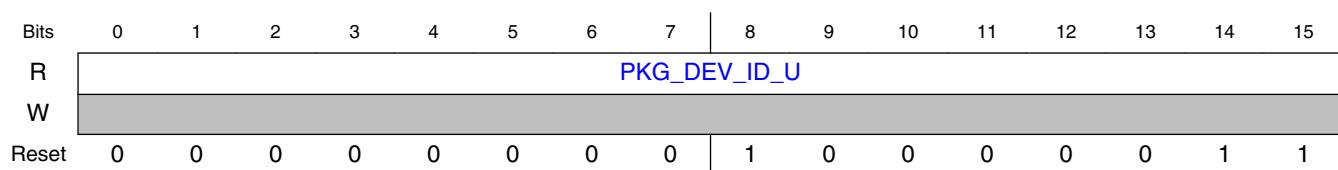
#### 33.6.6.8.1 Offset

Register	Offset
KX_AN_PKG_ID_H	Eh

#### 33.6.6.8.2 Function

The 1000Base-KX AN Package Device Identifier Upper Register contains the upper half of the 32-bit Package Identifier.

#### 33.6.6.8.3 Diagram



#### 33.6.6.8.4 Fields

Field	Function
0-15	Package Device Identifier Upper
PKG_DEV_ID_U	Package Device Identifier Upper: OUI[3:18]

### 33.6.6.9 KX AN Package Identifier Lower (KX\_AN\_PKG\_ID\_L)

#### 33.6.6.9.1 Offset

Register	Offset
KX_AN_PKG_ID_L	Fh

### 33.6.6.9.2 Function

The KX AN Package Identifier Lower Register contains the lower half of the 32-bit Package Identifier.

### 33.6.6.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	PKG_DEV_ID_L															
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0

### 33.6.6.9.4 Fields

Field	Function
0-15 PKG_DEV_ID_L	Package Device Identifier Lower Package Device Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

## 33.6.6.10 KX AN Advertisement 0 (KX\_AN\_ADVERT0)

### 33.6.6.10.1 Offset

Register	Offset
KX_AN_ADVERT0	10h

### 33.6.6.10.2 Function

The 1000Base-KX AN Advertisement Register 0 contains bits 15:0 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W	NP	ACK	REM_FAULT	XNP_CAP	PAUSE_CAP		ECHOED_NONCE					SELECT_FLD	D			
Reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1

### 33.6.6.10.4 Fields

Field	Function
0 NP	Next Page Next Page 0b - The device has no next pages to send 1b - The device has next pages to send
1 ACK	Ack Acknowledge Should always be set to 0
2 REM_FAULT	Remote Fault Remote Fault 0b - No remote fault advertised 1b - Advertise remote fault
3 XNP_CAP	Extended Next Page Capabilty Extended Next Page Capability 0b - Device does not support extended next pages
4-5 PAUSE_CAP	Pause Capability Pause Capability (ASM_DIR:PAUSE) Default is 11 00b - No PAUSE 01b - Symmetric PAUSE 10b - Asymmetric PAUSE toward link partner 11b - Both Symmetric PAUSE and Asymmetric PAUSE toward local device
6-10 ECHOED_NONCE	Echoed Nonce Echoed Nonce field (E4:0) Contains the transmitted nonce field from the link partner Only valid if ACK is set
11-15 SELECT_FLD	Selector field Selector field (S4:0) All values not shown are reserved 00001b - IEEE Std 802.3

### 33.6.6.11 KX AN Advertisement 1 (KX\_AN\_ADVERT1)

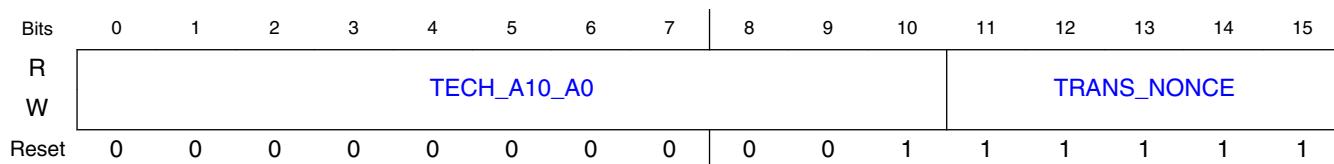
#### 33.6.6.11.1 Offset

Register	Offset
KX_AN_ADVERT1	11h

#### 33.6.6.11.2 Function

The 1000Base-KX AN Advertisement Register 1 contains bits 31:16 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2.

#### 33.6.6.11.3 Diagram



#### 33.6.6.11.4 Fields

Field	Function
0-10 TECH_A10_A0	Technology A10:A0 Bits A10:A0 of the technology field. 5 (A0): 1000Base-KX 6 (A1): 10GBase-KX4 7 (A2): reserved 8 (A3): reserved 9 (A4): reserved 10 (A5): reserved 11 (A6): reserved 12 (A7): reserved 13 (A8): reserved 14 (A9): reserved 15 (A10): reserved

*Table continues on the next page...*

## MDIO register spaces

Field	Function
	Must set all bits except A0 to 0
11-15	TransmittedNonce
TRANS_NONCE	TransmittedNonce
E	Two devices must have a different nonce for auto-negotiation to operate (i.e. a loopback will not allow auto-negotiation to complete.)

### 33.6.6.12 KX AN Advertisement 2 (KX\_AN\_ADVERT2)

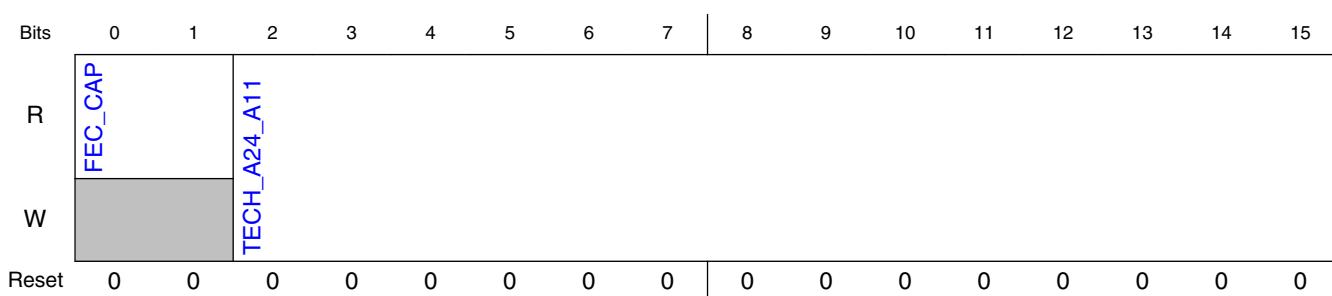
#### 33.6.6.12.1 Offset

Register	Offset
KX_AN_ADVERT2	12h

#### 33.6.6.12.2 Function

The 1000Base-KX AN Advertisement Register 0 contains bits 47:32 of the 48-bit page bits used during base page exchange. They should be programmed before auto-negotiation is started. The 48-bit value is distributed over registers 0, 1, 2. Register 2 must be written last.

#### 33.6.6.12.3 Diagram



#### 33.6.6.12.4 Fields

Field	Function
0-1	FEC Capability
FEC_CAP	Capability bits (F1:0) for FEC support

Table continues on the next page...

Field	Function
	00b - PCS does not implement a FEC function
2-15	Technology A24:A11
TECH_A24_A11	Technology field A24:11
	All bits reserved. Should be set to 0

### 33.6.6.13 KX AN LP Base Page Ability 0 (KX\_AN\_LP\_BASE\_PG\_ABI\_L0)

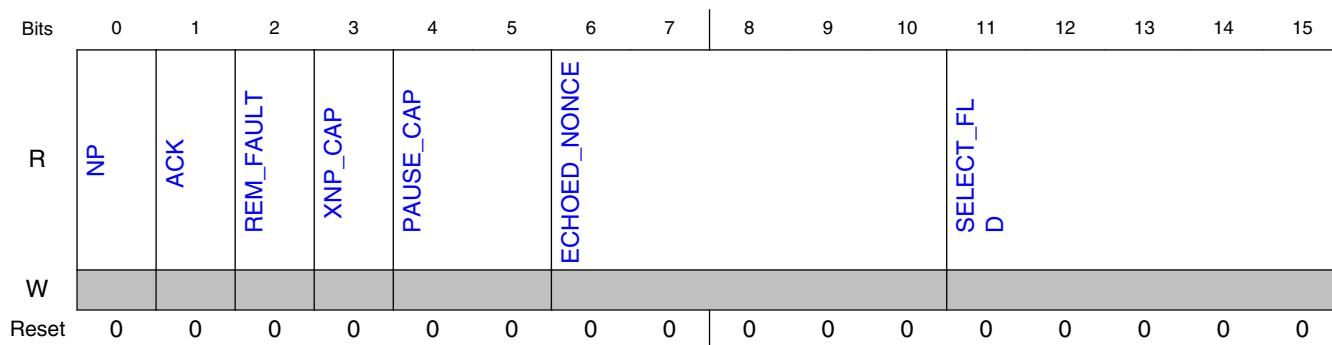
#### 33.6.6.13.1 Offset

Register	Offset
KX_AN_LP_BASE_PG_ABI0	13h

#### 33.6.6.13.2 Function

The 1000Base-KX AN LP Base Page Ability Register 0 contains bits 15:0 of the link partner base storage. The 48-bit value is stored over registers 0, 1 and 2.

#### 33.6.6.13.3 Diagram



#### 33.6.6.13.4 Fields

Field	Function
0	Next Page
NP	Next Page

Table continues on the next page...

## MDIO register spaces

Field	Function
	0b - The link partner has no next pages to send 1b - The link partner has next pages to send
1 ACK	Ack Acknowledge 0b - No link partner acknowledge 1b - Link partner acknowledge
2 REM_FAULT	Remote Fault Remote Fault 0b - No remote fault advertised by link partner 1b - Remote fault advertised by link partner
3 XNP_CAP	Extended Next Page Capability Extended Next Page Capability 0b - Link partner does not support extended next pages 1b - Link parnet supports extended next pages
4-5 PAUSE_CAP	Pause capability Pause Capability (ASM_DIR:PAUSE) 00b - No PAUSE 01b - Symmetric PAUSE 10b - Asymmetric PAUSE toward local device 11b - Both Symmetric PAUSE and Asymmetric PAUSE toward link partner
6-10 ECHOED_NON CE	Echoed nonce Echoed Nonce field (E4:0) Contains the transmitted nonce field from the device as seen by the link partner Only valid if ACK is set
11-15 SELECT_FLD	Selector field Selector field (S4:0) All values not shown are reserved 00001b - IEEE Std 802.3

## 33.6.6.14 KX AN LP Base Page Ability 1 (KX\_AN\_LP\_BASE\_PG\_ABI\_L1)

### 33.6.6.14.1 Offset

Register	Offset
KX_AN_LP_BASE_PG_ABI1	14h

### 33.6.6.14.2 Function

The 1000Base-KX AN LP Base Page Ability Register 1 contains bits 31:16 of the link partner base storage. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TECH_A10_A0								TRANS_NONCE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.6.14.4 Fields

Field	Function
0-10 TECH_A10_A0	Technology A10:A0 Bits A10:A0 of the technology field. 5 (A0): 1000Base-KX 6 (A1): 10GBase-KX4 7 (A2): reserved 8 (A3): reserved 9 (A4): reserved 10 (A5): reserved 11 (A6): reserved 12 (A7): reserved 13 (A8): reserved 14 (A9): reserved 15 (A10): reserved
11-15 TRANS_NONCE	Transmitted Nonce Transmitted Nonce

### 33.6.6.15 KX AN LP Base Page Ability 2 (KX\_AN\_LP\_BASE\_PG\_ABI\_L2)

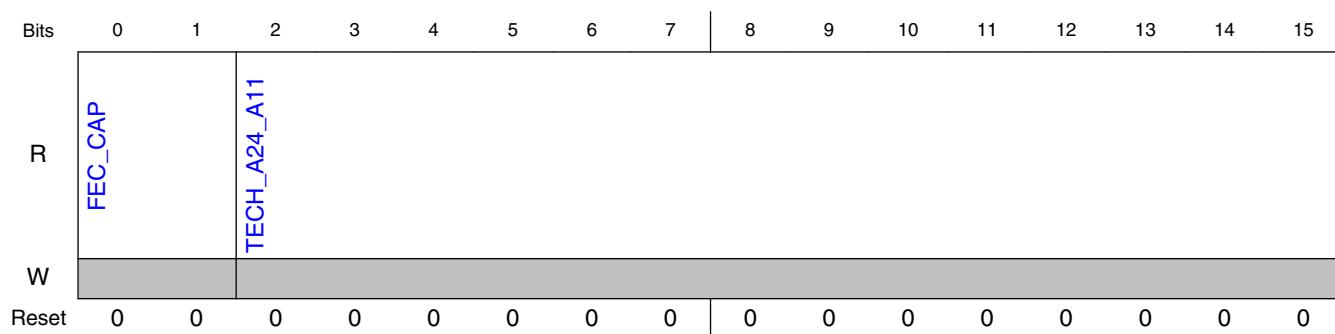
### 33.6.6.15.1 Offset

Register	Offset
KX_AN_LP_BASE_PG_ABIL2	15h

### 33.6.6.15.2 Function

The 1000Base-KX AN LP Base Page Ability Register 2 contains bits 47:32 of the link partner base storage. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.15.3 Diagram



### 33.6.6.15.4 Fields

Field	Function
0-1 FEC_CAP	FEC Capability Capability bits (F1:0) for FEC support F1: FEC ability F0: FEC requested
2-15 TECH_A24_A11	Technology A24:A11 Technology field A24:11 Reserved for future technology

## 33.6.6.16 KX AN XNP Transmit 0 (KX\_AN\_XNP\_TX0)

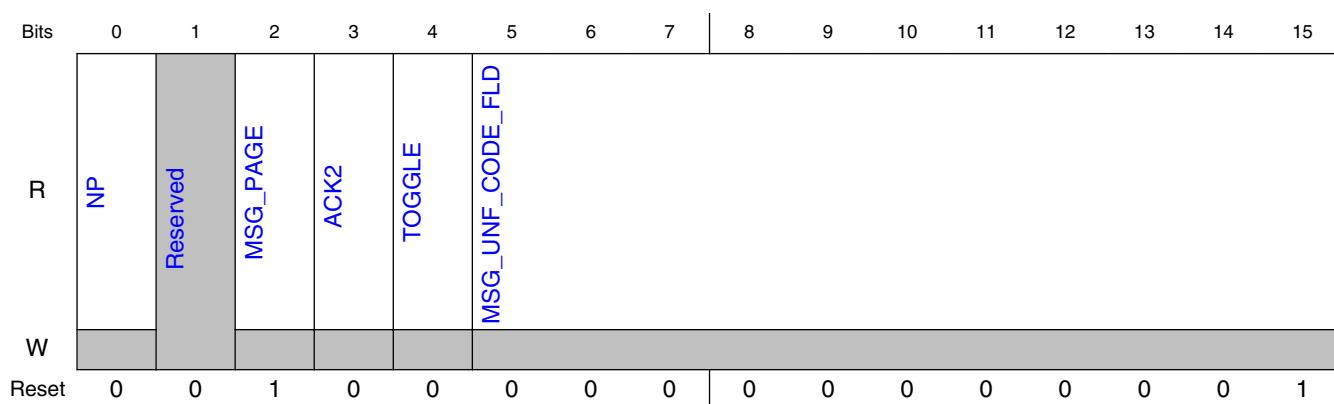
### 33.6.6.16.1 Offset

Register	Offset
KX_AN_XNP_TX0	16h

### 33.6.6.16.2 Function

The 1000Base-KX AN XNP Transmit Register 0 contains bits 15:0 of the XNP Transmit Register. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.16.3 Diagram



### 33.6.6.16.4 Fields

Field	Function
0 NP	Next Page Next Page 0b - Device does not have any more Next Pages to send 1b - Device has more Next Pages to Send
1 —	Reserved
2 MSG_PAGE	Message Page Message Page 0b - Next page is an unformatted page 1b - Next page is a message page
3 ACK2	Ack 2 Acknowledge 2 0b - The receiver is not able to act on the information defined in the message 1b - The receiver is able to act on the information defined in the message

Table continues on the next page...

## MDIO register spaces

Field	Function
4 TOGGLE	Toggle Toggle
5-15 MSG_UNF_CO DE_FLD	Message/Unformatted Code Field Message/Unformatted Code Field When MSG_PAGE=1: Message code field, else unformatted code field. For the null message code, the value is 0x1

## 33.6.6.17 KX AN XNP Transmit 1 (KX\_AN\_XNP\_TX1)

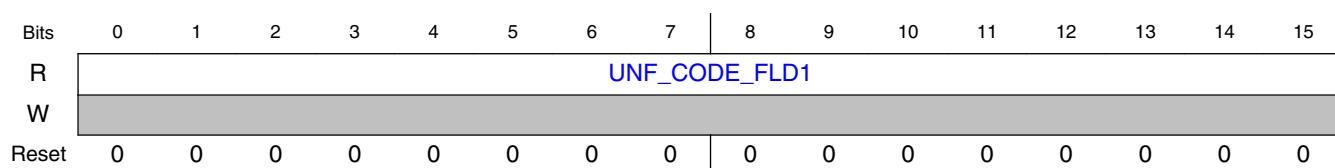
### 33.6.6.17.1 Offset

Register	Offset
KX_AN_XNP_TX1	17h

### 33.6.6.17.2 Function

The 1000Base-KX AN XNP Transmit Register 1 contains bits 31:16 of the XNP Transmit Register. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.17.3 Diagram



### 33.6.6.17.4 Fields

Field	Function
0-15 UNF_CODE_FL D1	Unformatted Code Field 1 Unformatted Code Field 1

### 33.6.6.18 KX AN XNP Transmit 2 (KX\_AN\_XNP\_TX2)

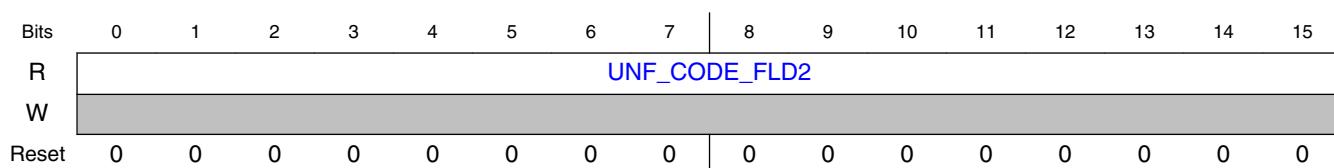
#### 33.6.6.18.1 Offset

Register	Offset
KX_AN_XNP_TX2	18h

#### 33.6.6.18.2 Function

The 1000Base-KX AN XNP Transmit Register 2 contains bits 48:32 of the XNP Transmit Register. The 48-bit value is distributed over registers 0, 1, 2. Register 2 must be written last.

#### 33.6.6.18.3 Diagram



#### 33.6.6.18.4 Fields

Field	Function
0-15	Unformatted Code Field 2
UNF_CODE_FLD2	Unformatted Code Field 2

### 33.6.6.19 KX AN LP XNP Ability 0 (KX\_AN\_LP\_XNP\_ABIL0)

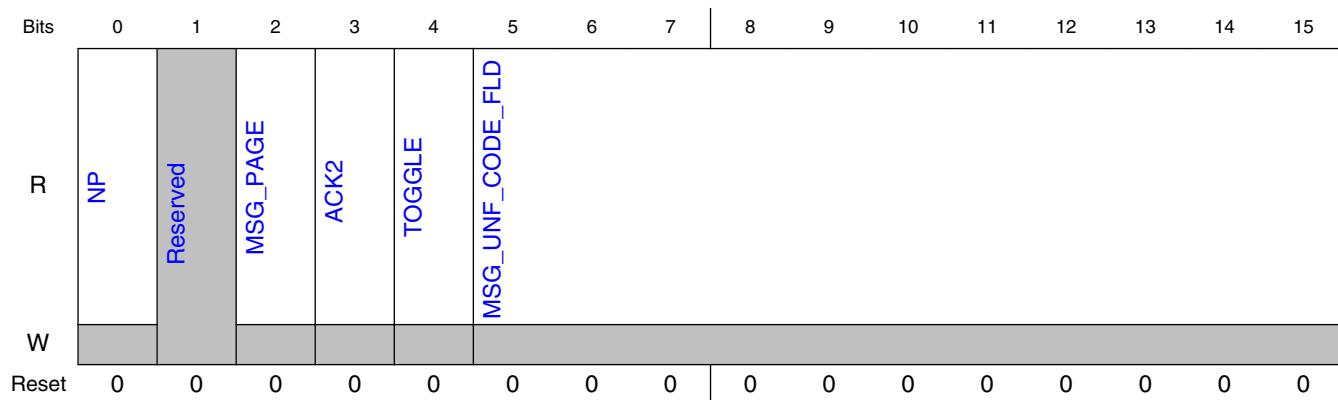
#### 33.6.6.19.1 Offset

Register	Offset
KX_AN_LP_XNP_ABIL0	19h

### 33.6.6.19.2 Function

The 1000Base-KX AN LP XNP Ability Register 0 contains bits 15:0 of the LP XNP Ability Register. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.19.3 Diagram



### 33.6.6.19.4 Fields

Field	Function
0 NP	Next Page Next Page 0b - Device does not have any more Next Pages to send 1b - Device has more Next Pages to Send
1 —	Reserved
2 MSG_PAGE	Message Page Message Page 0b - Next page is an unformatted page 1b - Next page is a message page
3 ACK2	Ack 2 Acknowledge 2 0b - The receiver is not able to act on the information defined in the message 1b - The receiver is able to act on the information defined in the message
4 TOGGLE	Toggle Toggle
5-15 MSG_UNF_CODE_FLD	Message/Unformatted Code Field Message/Unformatted Code Field When MSG_PAGE=1: Message code field, else unformatted code field. For the null message code, the value is 0x1

### 33.6.6.20 KX AN LP XNP Ability 1 (KX\_AN\_LP\_XNP\_ABIL1)

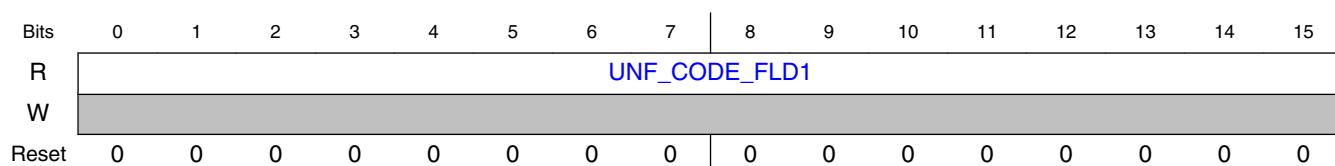
#### 33.6.6.20.1 Offset

Register	Offset
KX_AN_LP_XNP_ABIL1	1Ah

#### 33.6.6.20.2 Function

The 1000Base-KX AN LP XNP Ability Register 1 contains bits 31:16 of the XNP Transmit Register. The 48-bit value is distributed over registers 0, 1, 2.

#### 33.6.6.20.3 Diagram



#### 33.6.6.20.4 Fields

Field	Function
0-15	Unformatted Code Field 1
UNF_CODE_FL D1	Unformatted Code Field 1

### 33.6.6.21 KX AN LP XNP Ability 2 (KX\_AN\_LP\_XNP\_ABIL2)

#### 33.6.6.21.1 Offset

Register	Offset
KX_AN_LP_XNP_ABIL2	1Bh

### 33.6.6.21.2 Function

The 1000Base-KX AN LP XNP Ability Register 2 contains bits 48:32 of the XNP Transmit Register. The 48-bit value is distributed over registers 0, 1, 2.

### 33.6.6.21.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									UNF_CODE_FLD2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.6.21.4 Fields

Field	Function
0-15	Unformatted Code Field 2
UNF_CODE_FL D2	Unformatted Code Field 2

## 33.6.6.22 KX Backplane Ethernet Status (KX\_BP\_STAT)

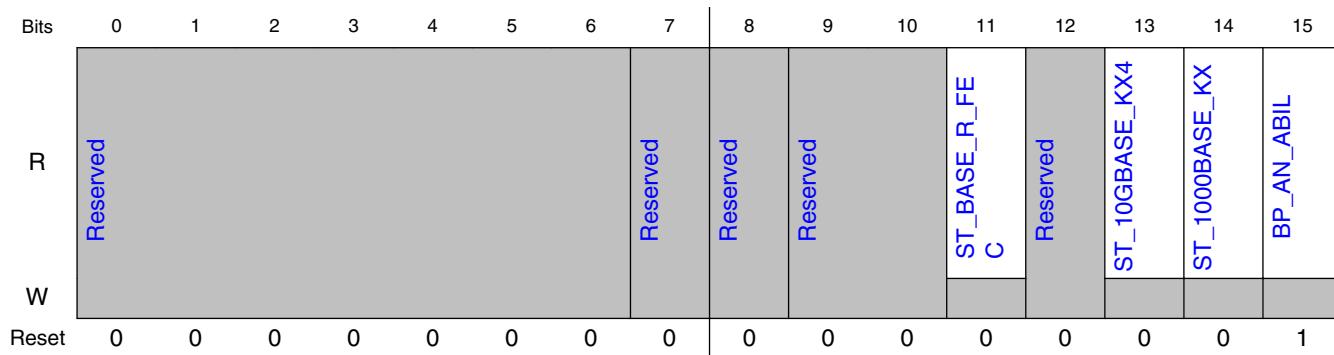
### 33.6.6.22.1 Offset

Register	Offset
KX_BP_STAT	30h

### 33.6.6.22.2 Function

The 1000Base-KX AN Backplane Ethernet Status Register provides the result after auto-negotiation DME page exchange completed.

### 33.6.6.22.3 Diagram



### 33.6.6.22.4 Fields

Field	Function
0-6 —	Reserved
7 —	Reserved
8 —	Reserved
9-10 —	Reserved
11 ST_BASE_R_FEC	Base-R FEC Always 0 after negotiation complete
12 —	Reserved
13 ST_10GBASE_KX4	10GBase-KX4 Always 0 after negotiation complete
14 ST_1000BASE_KX	1000Base-KX 1000Base-KX 0b - Not auto-negotiated to 1000Base-KX 1b - Auto-negotiated to 1000Base-KX
15 BP_AN_ABIL	Backplane AN Ability 1000Base-KX capable PHY type is implemented Always 1

### 33.6.6.23 KX Millisecond Count (KX\_MS\_CNT)

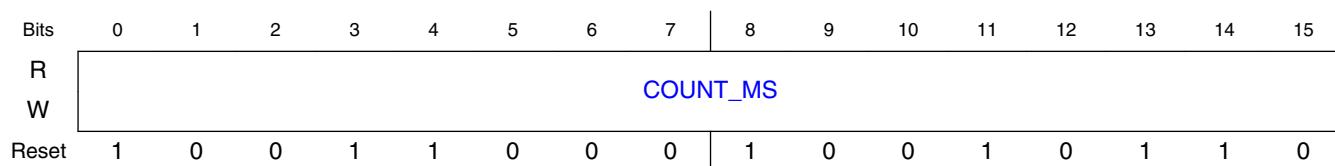
#### 33.6.6.23.1 Offset

Register	Offset
KX_MS_CNT	8000h

#### 33.6.6.23.2 Function

The 1000Base-KX AN Millisecond Counter Register contains the upper 16 bits of the 18-bit counter value for counting 1 ms.

#### 33.6.6.23.3 Diagram



#### 33.6.6.23.4 Fields

Field	Function
0-15	Count Milliseconds
COUNT_MS	Count milliseconds Upper 16-bits value of the 18-bit counter is provided for counting 1ms. The milliseconds counter operates on 8bit samples (i.e. 6.4ns) incrementing by 4 with every sample (i.e. $0x9896 * 4 * 6.4ns = 1ms$ )

### 33.6.7 MDIO\_KX\_VENDOR\_SPEC register descriptions

The 1000Base-KX vendor-specific 1 register space is selected when the associated SGMIIInCR1[MDEV\_PORT] matches the Ethernet MAC port address (MDIO\_CTL[PORT\_ADDR]) and the device address (MDIO\_CTL[DEV\_ADDR]) is 1Dh.

### 33.6.7.1 MDIO\_KX\_VENDOR\_SPEC memory map

MDIO\_KX\_VENDOR\_SPEC base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	KX Revision (KX_VND_REV)	16	RO	0114h
1h	KX Scratch (KX_VND_SCRATCH)	16	RW	0000h
2h	KX PCS Interrupt Event (KX_VND_PCS_INT)	16	RW	0000h
3h	KX PCS Interrupt Mask (KX_VND_INT_MSK)	16	RW	0000h
4h	KX AN Interrupt Event (KX_VND_AN_INT)	16	RW	0000h
5h	KX AN Interrupt Mask (KX_VND_AN_INT_MSK)	16	RW	0000h

### 33.6.7.2 KX Revision (KX\_VND\_REV)

#### 33.6.7.2.1 Offset

Register	Offset
KX_VND_REV	0h

#### 33.6.7.2.2 Function

The 1000Base-KX version register contains version information for the KX PHY.

#### 33.6.7.2.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
	CFG_VER								IP_VER			IP_REV				
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0

#### 33.6.7.2.4 Fields

Field	Function
0-7	Configuration/Integration Version
CFG_VER	Configuration/Integration Version

Table continues on the next page...

## MDIO register spaces

Field	Function
8-11	IP Version
IP_VER	IP Version
12-15	IP revision
IP_REV	IP Revision

### 33.6.7.3 KX Scratch (KX\_VND\_SCRATCH)

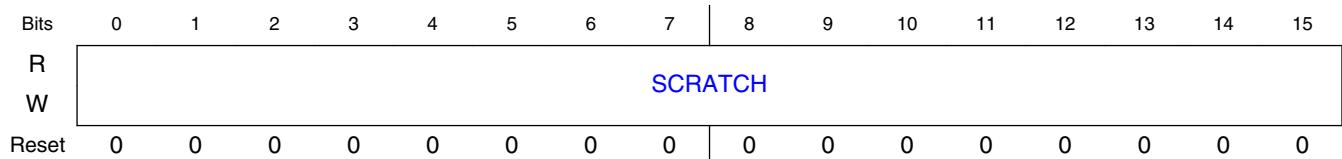
#### 33.6.7.3.1 Offset

Register	Offset
KX_VND_SCRATCH	1h

#### 33.6.7.3.2 Function

The 1000Base-KX scratch register may be used to test register reads and writes.

#### 33.6.7.3.3 Diagram



#### 33.6.7.3.4 Fields

Field	Function
0-15	Scratch
SCRATCH	Scratch register

### 33.6.7.4 KX PCS Interrupt Event (KX\_VND\_PCS\_INT)

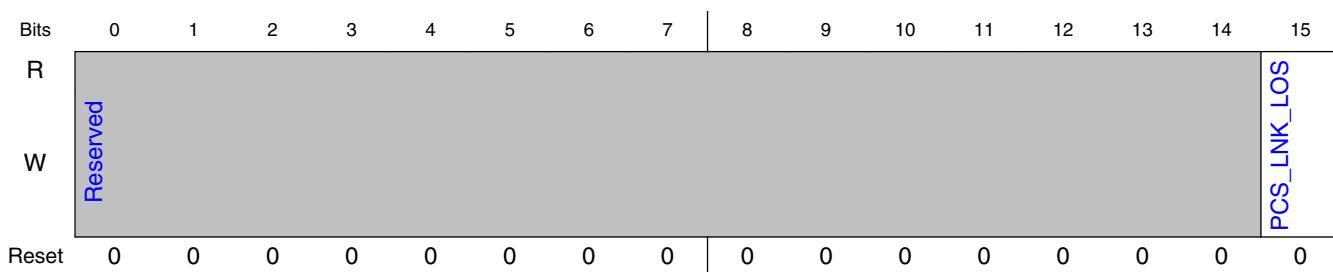
### 33.6.7.4.1 Offset

Register	Offset
KX_VND_PCS_INT	2h

### 33.6.7.4.2 Function

The 1000Base-KX PCS interrupt event register contains detect bits for KX PCS interrupt events.

### 33.6.7.4.3 Diagram



### 33.6.7.4.4 Fields

Field	Function
0-14	Reserved
—	
15 PCS_LNK_LOS	PCS Link LOS PCS Link Loss Cleared on register read  0b - PCS has not detected a loss of link synchronization 1b - PCS has detected a loss of link synchronization

## 33.6.7.5 KX PCS Interrupt Mask (KX\_VND\_INT\_MSK)

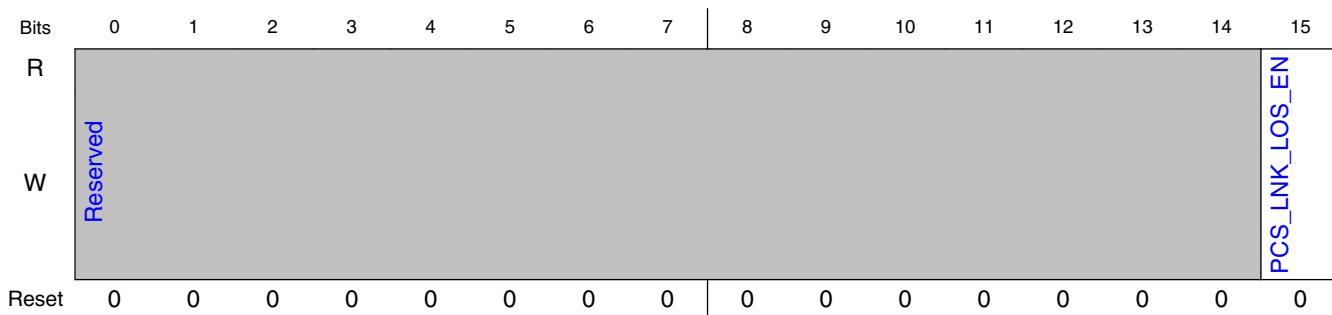
### 33.6.7.5.1 Offset

Register	Offset
KX_VND_INT_MSK	3h

### 33.6.7.5.2 Function

The 1000Base-KX PCS Interrupt Event Register contains mask bits for KX PCS interrupt events. See the Frame Manager reference manual for details on handling of PCS interrupts.

### 33.6.7.5.3 Diagram



### 33.6.7.5.4 Fields

Field	Function
0-14 —	Reserved
15 PCS_LNK_LOS_EN	PCS Link LOS Enable PCS Link Loss Event Enable 0b - A PCS loss of link synchronization will not cause an interrupt event

## 33.6.7.6 KX AN Interrupt Event (KX\_VND\_AN\_INT)

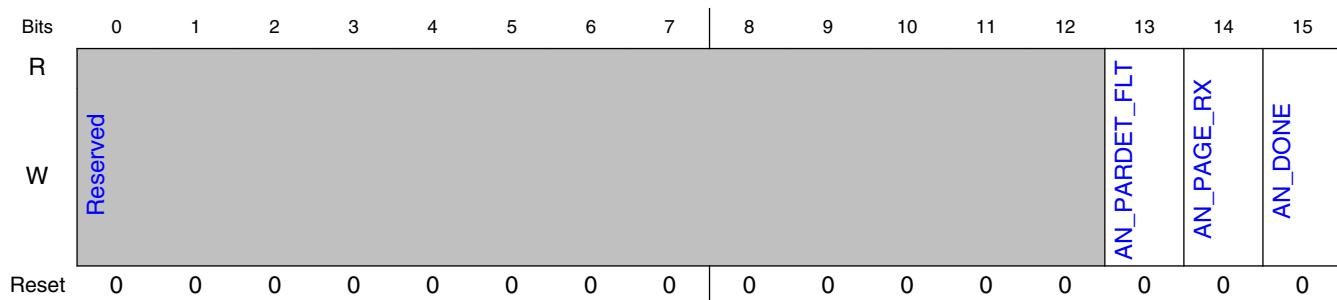
### 33.6.7.6.1 Offset

Register	Offset
KX_VND_AN_INT	4h

### 33.6.7.6.2 Function

The 1000Base-KX PCS Interrupt Event Register contains detect bits for KX auto-negotiation interrupt events.

### 33.6.7.6.3 Diagram



### 33.6.7.6.4 Fields

Field	Function
0-12 —	Reserved
13 AN_PARDET_F LT	AN Parallel Detection Fault Parallel detection fault Cleared on register read 0b - No fault detected 1b - Ambiguous link status detected while waiting on DME page receive
14 AN_PAGE_RX	AN Page Rx Auto-negotiate page receive Indicates the validity of the remote ability word (base page or next page) Cleared on register read 0b - An ability word was not received from the remote device 1b - An ability word was received from the remote device during backplane auto-negotiation
15 AN_DONE	AN Done Backplane auto-negotiation complete Note: not used for 1000Base-X/SGMII auto-negotiation Cleared on register read 0b - Backplane auto-negotiation not complete 1b - Backplane auto-negotiation complete

### 33.6.7.7 KX AN Interrupt Mask (KX\_VND\_AN\_INT\_MSK)

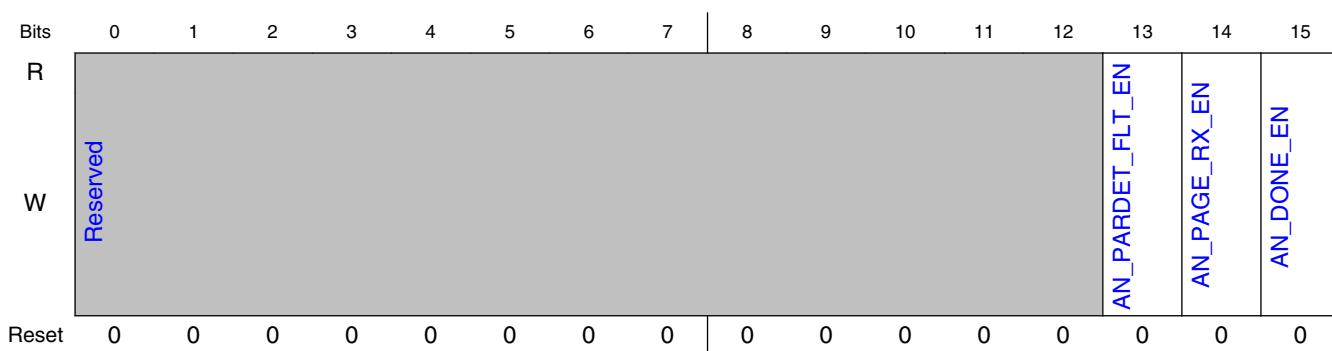
#### 33.6.7.7.1 Offset

Register	Offset
KX_VND_AN_INT_MSK	5h

#### 33.6.7.7.2 Function

The 1000Base-KX PCS Interrupt Event Register contains mask bits for KX AN interrupt events. See the Frame Manager reference manual for details on handling of AN interrupts.

#### 33.6.7.7.3 Diagram



#### 33.6.7.7.4 Fields

Field	Function
0-12	Reserved
—	
13 AN_PARDET_FLT_EN	AN Parallel Fault Enable AN Parallel Detection Fault Event Enable 0b - AN Parallel Detection Fault will not cause an interrupt event 1b - AN Parallel Detection Fault will cause an interrupt event
14 AN_PAGE_RX_EN	AN Page Rx Enable AN Page Receive Event Enable 0b - AN Page Receive will not cause an interrupt event 1b - AN Page Receive will cause an interrupt event
15	AN Done Enable

Field	Function
AN_DONE_EN	AN Done Event Enable 0b - AN done will not cause an interrupt event 1b - AN done will cause an interrupt event

## 33.6.8 MDIO\_SGMII register descriptions

The SGMII MDIO register space is selected when the associated SGMIIInCR1[MDEV\_PORT] matches the Ethernet MAC PHY address (MDIO\_CTL[PHY\_ADDR]).

### 33.6.8.1 MDIO\_SGMII memory map

MDIO\_SGMII base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	SGMII Control (SGMII_CR)	16	RW	1140h
1h	SGMII Status (SGMII_SR)	16	RO	0009h
2h	SGMII PHY Identifier Upper (SGMII_PHY_ID_H)	16	RO	0083h
3h	SGMII PHY Identifier Lower (SGMII_PHY_ID_L)	16	RO	E400h
4h	SGMII Device Ability for 1000Base-X (SGMII_DEV_ABIL_1KBX)	16	RW	01A0h
4h	SGMII Device Ability for SGMII (SGMII_DEV_ABIL_SGMII)	16	RW	01A0h
5h	SGMII Partner Ability for 1000Base-X (SGMII_LP_DEV_ABIL_1KBX)	16	RO	0000h
5h	SGMII Partner Ability for SGMII (SGMII_LP_DEV_ABIL_SGMII)	16	RO	0000h
6h	SGMII AN Expansion (SGMII_AN_EXP)	16	RO	0004h
7h	SGMII Next Page Transmit (SGMII_NP_TX)	16	RW	0000h
8h	SGMII LP Next Page Receive (SGMII_NP_RX)	16	RO	0000h
Fh	SGMII Extended Status (SGMII_XTND_STAT)	16	RU	0000h
10h	SGMII Scratch (SGMII_SCRATCH)	16	RW	0000h
11h	SGMII Design Revision (SGMII_REV)	16	RO	0001h
12h	SGMII Link Timer Lower (SGMII_LINK_TMR_L)	16	RW	12D0h
13h	SGMII Link Timer Upper (SGMII_LINK_TMR_H)	16	RW	0013h
14h	SGMII IF Mode (SGMII_IF_MODE)	16	RW	0000h

## 33.6.8.2 SGMII Control (SGMII\_CR)

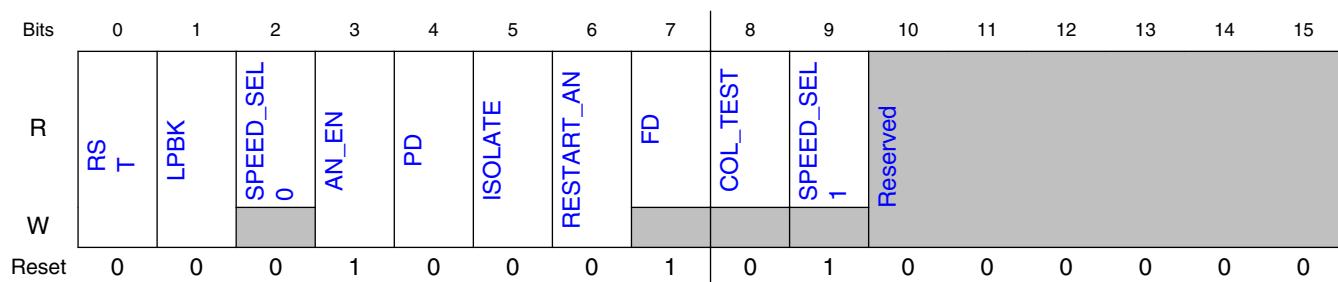
### 33.6.8.2.1 Offset

Register	Offset
SGMII_CR	0h

### 33.6.8.2.2 Function

The SGMII Control Register contains control bits used to enable/disable functions in the PCS, and to initiate commands such as reset.

### 33.6.8.2.3 Diagram



### 33.6.8.2.4 Fields

Field	Function
0 RST	Reset Self-Clearing Read / Write Reset Command Register. When set to 1, a synchronous reset pulse is generated which resets all the PCS state machines, the Comma detection function and the 8b/10b coder / decoder. Should be set to 0 (default) for normal operation.
1 LPBK	Loopback Loopback Command. 0b - Normal operation 1b - Reserved
2 SPEED_SEL0	Speed selection (LSB) Speed selection (LSB) All others reserved Note: SGMII speed is controlled with the IF Mode register MSB,LSB 1,0: 1000 Mb/s

Table continues on the next page...

Field	Function
	All others reserved
3 AN_EN	AN Enable Auto-negotiation enable 0b - Disable auto-negotiation 1b - Enable auto-negotiation
4 PD	Power down Power down 0b - Normal operation 1b - Power down PCS. The PCS is held in internal soft reset until the bit is cleared again.
5 ISOLATE	Isolate Isolate 0b - Normal operation 1b - Reserved
6 RESTART_AN	Restart AN Restart auto-negotiation This bit is self-clearing 0b - Normal operation 1b - Restart auto-negotiation
7 FD	Full Duplex Duplex mode. Read-only 0b - Reserved 1b - Full duplex
8 COL_TEST	Collision test Collision test. Read-only 0b - Disable COL signal test 1b - Reserved
9 SPEED_SEL1	Speed selection (MSB) Speed selection (MSB) See SPEED_SEL0
10-15 —	Reserved

### 33.6.8.3 SGMII Status (SGMII\_SR)

#### 33.6.8.3.1 Offset

Register	Offset
SGMII_SR	1h

### 33.6.8.3.2 Function

The SGMII Status Register contains status bits on the operation of the PCS.

### 33.6.8.3.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ST_100BASET4	FD_100BASEX	HD_100BASEX	FD_10MBPS	HD_10MBPS	FD_100BASET2	HD_100BASET2	EXT_STAT	Reserved	AN_COMP	REM_FAULT	AN_ABIL	LINK_STAT	JAB_DET	EXT_CAP	
W									0	0	0	0	1	0	1	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	

### 33.6.8.3.4 Fields

Field	Function
0 ST_100BASET4	100Base-T4 Read Only bit set to 0 to indicate that the PCS does not support 100Base-T4 operation
1 FD_100BASEX	100Base-X Full Duplex Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
2 HD_100BASEX	100Base-X Half Duplex Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
3 FD_10MBPS	10Mbps Full Duplex Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
4 HD_10MBPS	10Mbps Half Duplex Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
5 FD_100BASET2	100Base-T2 Full Duplex Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
6 HD_100BASET2	100Base-T2 Half Duplex Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
7 EXT_STAT	Extended Status Read Only bit always set to 0 to indicate that the PCS does not implement an extended status register.
8-9 —	Reserved
10 AN_COMP	AN Complete Auto-negotiation complete. Read Only Bit

Table continues on the next page...

Field	Function
	0b - The Auto Negotiation process is not completed or Auto Negotiation is disabled 1b - The Auto Negotiation process is completed and that the Auto Negotiation control registers are valid.
11 REM_FAULT	Remote Fault Read Only Bit always set to 0. The PCS does not implement a PHY specific remote fault detection optional function.
12 AN_ABIL	AN Ability Auto Negotiation Ability. Read Only Bit set to „1. to indicate that the PCS supports Auto-Negotiation.
13 LINK_STAT	Link Status Link Status 0b - The link is not valid. If the link synchronization is lost a 0 is latched which is cleared only after a register read access 1b - This link is valid.
14 JAB_DET	Jabber Detection Read Only bit always set to 0, the Core does not support the optional Jabber detection function
15 EXT_CAP	Extended Capability Read Only bit set to „1. to indicate that the Core supports extended registers

### 33.6.8.4 SGMII PHY Identifier Upper (SGMII\_PHY\_ID\_H)

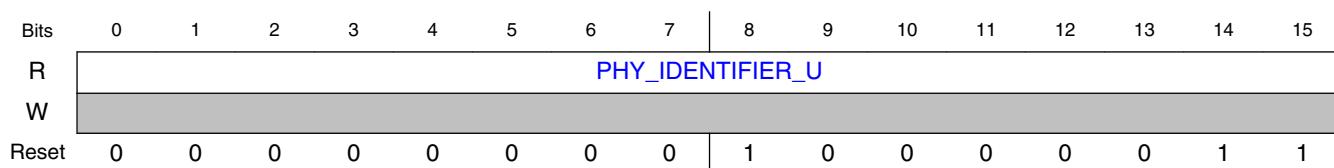
#### 33.6.8.4.1 Offset

Register	Offset
SGMII_PHY_ID_H	2h

#### 33.6.8.4.2 Function

The SGMII PHY Identifier Upper Register contains the upper half of the 32-bit PHY Identifier.

#### 33.6.8.4.3 Diagram



### 33.6.8.4.4 Fields

Field	Function
0-15	PHY Identifier Upper
PHY_IDENTIFIER_U	PHY Identifier Upper: OUI[3:18]

### 33.6.8.5 SGMII PHY Identifier Lower (SGMII\_PHY\_ID\_L)

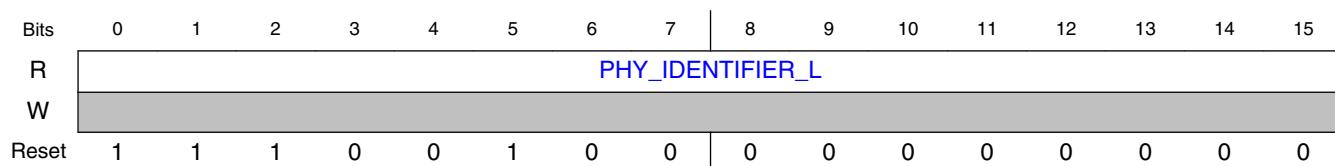
#### 33.6.8.5.1 Offset

Register	Offset
SGMII_PHY_ID_L	3h

#### 33.6.8.5.2 Function

The SGMII PHY Identifier Lower Register contains the lower half of the 32-bit PHY Identifier.

#### 33.6.8.5.3 Diagram



#### 33.6.8.5.4 Fields

Field	Function
0-15	PHY Identifier Lower
PHY_IDENTIFIER_L	PHY Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

### 33.6.8.6 SGMII Device Ability for 1000Base-X (SGMII\_DEV\_ABIL\_1KBX)

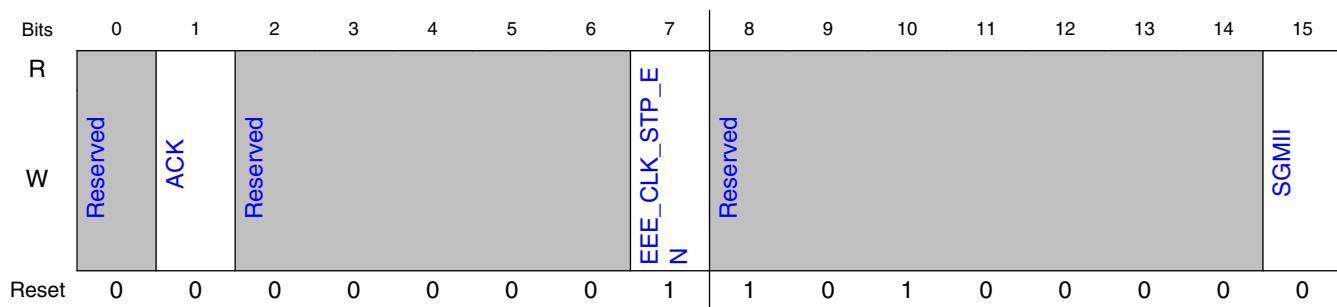
#### 33.6.8.6.1 Offset

Register	Offset
SGMII_DEV_ABIL_1KBX	4h

#### 33.6.8.6.2 Function

The SGMII Device Ability Register contains the control bits used to advertise the device abilities to the Link Partner during auto-negotiation for 1000Base-X mode.

#### 33.6.8.6.3 Diagram



#### 33.6.8.6.4 Fields

Field	Function
0	Reserved
—	
1	Ack
ACK	Read only bit set to 1 when device has received three consecutive matching ability values from the link partner
2-6	Reserved. Should be set to 0
—	
7	EEE Clock Stop Enable
EEE_CLK_STP_EN	EEE Clock Stop Enable

Table continues on the next page...

## MDIO register spaces

Field	Function
	Should be set to 0 before SGMII Auto-Negotiation enable/restart, as this device does not support EEE clock stop. 0b - EEE Clock Stop Disabled 1b - EEE Clock Stop Enabled
8-14 —	Reserved. Should be set to 0
15 SGMII	SGMII mode. Should be set to 1.

## 33.6.8.7 SGMII Device Ability for SGMII (SGMII\_DEV\_ABIL\_SGMII)

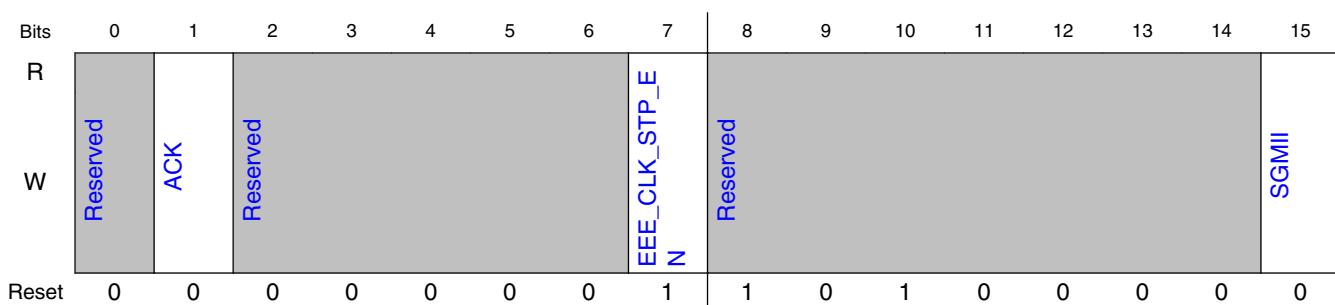
### 33.6.8.7.1 Offset

Register	Offset
SGMII_DEV_ABIL_SGMII	4h

### 33.6.8.7.2 Function

The SGMII Device Ability Register contains the control bits used to advertise the device abilities to the Link Partner during auto-negotiation for SGMII mode.

### 33.6.8.7.3 Diagram



### 33.6.8.7.4 Fields

Field	Function
0	Reserved

Table continues on the next page...

Field	Function
—	
1 ACK	Ack Read only bit set to 1 when device has received three consecutive matching ability values from the link partner
2-6	Reserved. Should be set to 0
—	
7 EEE_CLK_STP _EN	EEE Clock Stop Enable EEE Clock Stop Enable Should be set to 0 before SGMII Auto-Negotiation enable/restart, as this device does not support EEE clock stop. 0b - EEE Clock Stop Disabled 1b - EEE Clock Stop Enabled
8-14	Reserved. Should be set to 0
—	
15 SGMII	SGMII SGMII mode. Should be set to 1.

### 33.6.8.8 SGMII Partner Ability for 1000Base-X (SGMII\_LP\_DEV\_ABIL\_1KBX)

#### 33.6.8.8.1 Offset

Register	Offset
SGMII_LP_DEV_ABIL_1KBX	5h

#### 33.6.8.8.2 Function

The SGMII Partner Ability Register contains the capability status of the 1000Base-X link partner.

### 33.6.8.8.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	<b>COP_LNK_STAT</b>	<b>ACK</b>	Reserved	<b>COP_DUPL</b>	<b>COP_SPD</b>		<b>EEE_CAP</b>	<b>EEE_CLK_STOP_CAP</b>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.8.8.4 Fields

Field	Function
0 COP_LNK_STA T	Copper Link Status Read only bit, used by the SGMII PHY to advertise the Link Partner Copper status: 0b - Copper interface link is down 1b - Copper interface link is up
1 ACK	Ack Read only bit set to 1. when the Link Partner Copper Interface advertises that it has received three consecutive matching ability values from the device
2 —	Always 0
3 COP_DUPL	Copper Duplex Read only bit, used by the SGMII PHY to advertise the Link Copper duplex capability: 0b - Copper Interface resolved to Half-Duplex 1b - Copper Interface resolved to Full-Duplex
4-5 COP_SPD	Copper Speed Read only bits, used by the SGMII PHY to advertise the Link Partner Copper interface speed 00b - Copper Interface Speed is 10Mbps 01b - Copper Interface Speed is 100Mbps 10b - Copper Interface Speed is Gigabit 11b - Reserved
6 EEE_CAP	EEE Capability EEE Capability 0b - EEE not supported 1b - EEE supported
7 EEE_CLK_STO P_CAP	EEE Clock Stop Capability EEE Clock Stop Capability 0b - EEE Clock Stop not supported 1b - EEE Clock Stop supported

Table continues on the next page...

Field	Function
8-15	Reserved
—	

### 33.6.8.9 SGMII Partner Ability for SGMII (SGMII\_LP\_DEV\_ABIL\_SGMII)

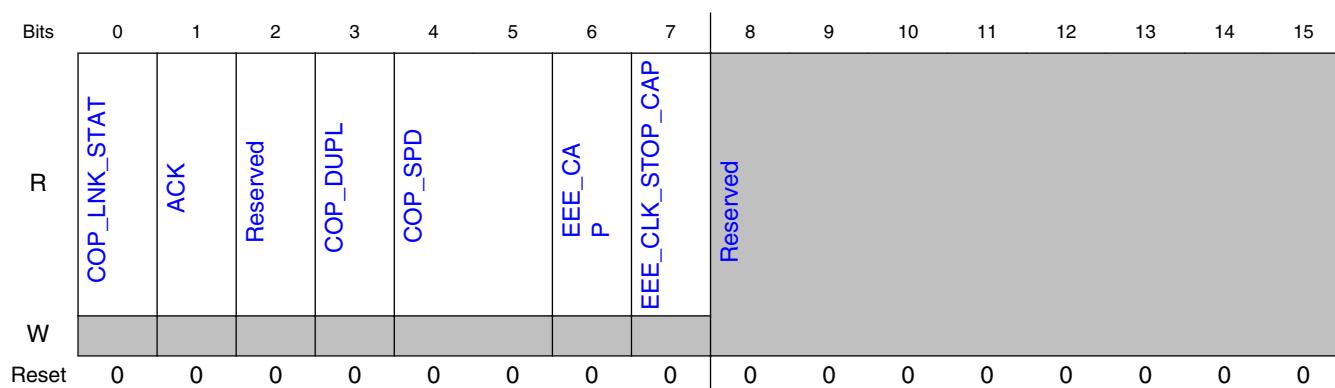
#### 33.6.8.9.1 Offset

Register	Offset
SGMII_LP_DEV_ABIL_SGMII	5h

#### 33.6.8.9.2 Function

The SGMII Partner Ability Register contains the capability status of the SGMII link partner.

#### 33.6.8.9.3 Diagram



#### 33.6.8.9.4 Fields

Field	Function
0 COP_LNK_STA T	Copper Link Status Read only bit, used by the SGMII PHY to advertise the Link Partner Copper status: 0b - Copper interface link is down

*Table continues on the next page...*

## MDIO register spaces

Field	Function
	1b - Copper interface link is up
1 ACK	Ack Read only bit set to 1. when the Link Partner Copper Interface advertises that it has received three consecutive matching ability values from the device
2 —	Always 0
3 COP_DUPL	Copper Duplex Read only bit, used by the SGMII PHY to advertise the Link Copper duplex capability: 0b - Copper Interface resolved to Half-Duplex 1b - Copper Interface resolved to Full-Duplex
4-5 COP_SPD	Copper Speed Read only bits, used by the SGMII PHY to advertise the Link Partner Copper interface speed 00b - Copper Interface Speed is 10Mbps 01b - Copper Interface Speed is 100Mbps 10b - Copper Interface Speed is Gigabit 11b - Reserved
6 EEE_CAP	EEE Capability EEE Capability 0b - EEE not supported 1b - EEE supported
7 EEE_CLK_STO P_CAP	EEE Clock Stop Capability EEE Clock Stop Capability 0b - EEE Clock Stop not supported 1b - EEE Clock Stop supported
8-15 —	Reserved

## 33.6.8.10 SGMII AN Expansion (SGMII\_AN\_EXP)

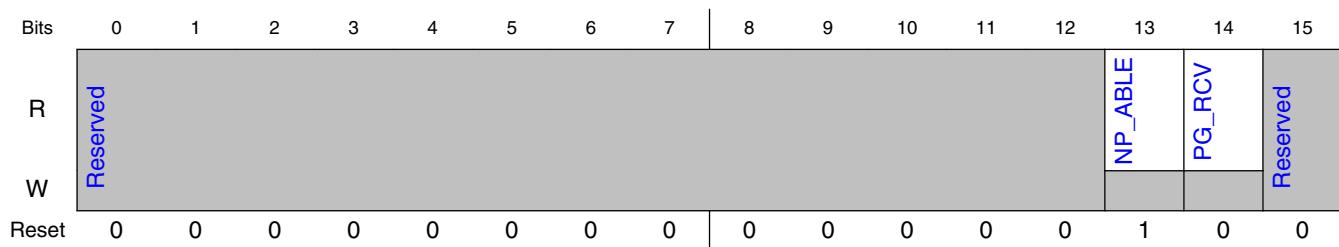
### 33.6.8.10.1 Offset

Register	Offset
SGMII_AN_EXP	6h

### 33.6.8.10.2 Function

The SGMII AN Expansion Register contains status bits indicating the PCS next page auto-negotiation capability and status.

### 33.6.8.10.3 Diagram



### 33.6.8.10.4 Fields

Field	Function
0-12 —	Reserved
13 NP_ABLE	Next Page Able Read Only bit set to 1 to indicate the PCS does support the Next Page function
14 PG_RCV	Page Received Set to 1 to indicate that a new page has been received with new partner ability available in the PCS register PARTNER_ABILITY. The bit is set to 0 (Reset value) when the system management agent performs a read access.
15 —	Reserved

### 33.6.8.11 SGMII Next Page Transmit (SGMII\_NP\_TX)

#### 33.6.8.11.1 Offset

Register	Offset
SGMII_NP_TX	7h

#### 33.6.8.11.2 Function

The SGMII NP TX Register contains next page data to transfer to the remote devide. Writing to this register initiates a next page exchange (sets mr\_np\_loaded variable).

### 33.6.8.11.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	NP	ACK	MP	ACK2	TOGGLE	DATA											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 33.6.8.11.4 Fields

Field	Function
0 NP	Next Page Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1 ACK	Ack Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2 MP	Message Page Message page (1) or unformatted page (0) format.
3 ACK2	Ack 2 Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4 TOGGLE	Toggle The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15 DATA	Data 11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

## 33.6.8.12 SGMII LP Next Page Receive (SGMII\_NP\_RX)

### 33.6.8.12.1 Offset

Register	Offset
SGMII_NP_RX	8h

### 33.6.8.12.2 Function

The SGMII NP RX Register contains the next page data received from the remote device during the latest next page exchange. The value is overridden with every new next page. Next page transfers are controlled by the application.

### 33.6.8.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	NP	ACK	MP	ACK2	TOGGLE	DATA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.8.12.4 Fields

Field	Function
0	Next Page
NP	Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1	Ack
ACK	Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2	Message Page
MP	Message page (1) or unformatted page (0) format.
3	Ack 2
ACK2	Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4	Toggle
TOGGLE	The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15	Data
DATA	11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

### 33.6.8.13 SGMII Extended Status (SGMII\_XTND\_STAT)

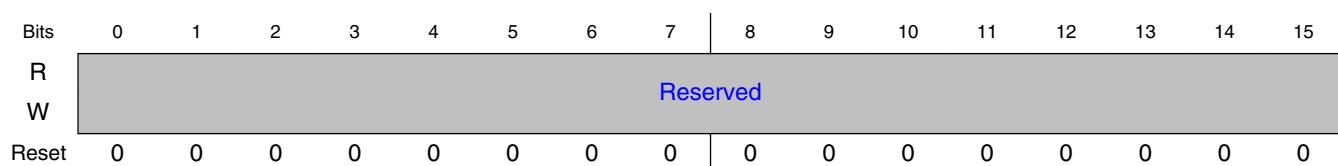
### 33.6.8.13.1 Offset

Register	Offset
SGMII_XTND_STAT	Fh

### 33.6.8.13.2 Function

The SGMII Extended Status Register is reserved, as this device does not implement the optional extended status registers.

### 33.6.8.13.3 Diagram



### 33.6.8.13.4 Fields

Field	Function
0-15	Reserved. Always 0
—	

## 33.6.8.14 SGMII Scratch (SGMII\_SCRATCH)

### 33.6.8.14.1 Offset

Register	Offset
SGMII_SCRATCH	10h

### 33.6.8.14.2 Function

The SGMII Scratch Register provides a memory location available to test register read and write operations.

### 33.6.8.14.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									SCRATCH							
W																

Reset 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0

### 33.6.8.14.4 Fields

Field	Function
0-15	Scratch
SCRATCH	Scratch field.

## 33.6.8.15 SGMII Design Revision (SGMII\_REV)

### 33.6.8.15.1 Offset

Register	Offset
SGMII_REV	11h

### 33.6.8.15.2 Function

The SGMII Revision Register contains revision information for the PCS.

### 33.6.8.15.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved			IP_MJ		IP_MN		
W																

Reset 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 1

### 33.6.8.15.4 Fields

Field	Function
0-7	Reserved

Table continues on the next page...

## MDIO register spaces

Field	Function
—	
8-11	Major Revision
IP_MJ	Major revision
12-15	Minor Revision
IP_MN	Minor revision

### 33.6.8.16 SGMII Link Timer Lower (SGMII\_LINK\_TMR\_L)

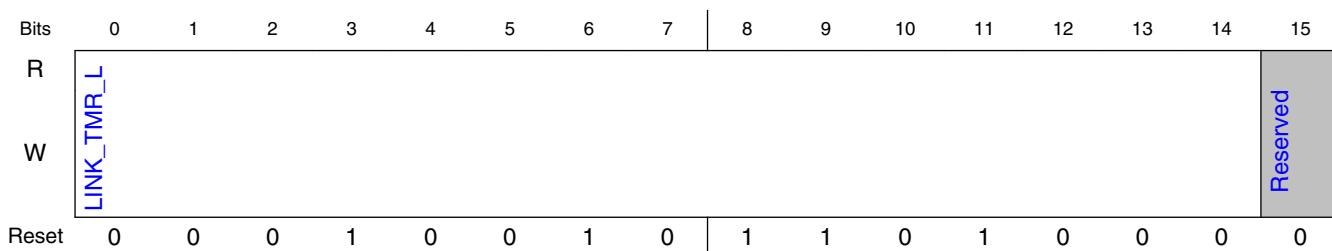
#### 33.6.8.16.1 Offset

Register	Offset
SGMII_LINK_TMR_L	12h

#### 33.6.8.16.2 Function

The SGMII Link Timer Register Lower contains bits 15:1 of the link timer value.

#### 33.6.8.16.3 Diagram



#### 33.6.8.16.4 Fields

Field	Function
0-14 LINK_TMR_L	Link Timer Lower Link timer[15:1]
15 —	Reserved

### 33.6.8.17 SGMII Link Timer Upper (SGMII\_LINK\_TMR\_H)

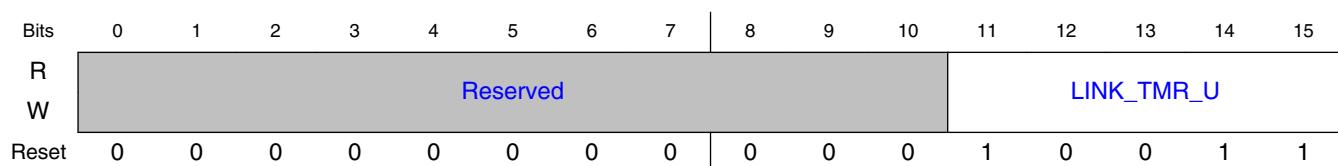
#### 33.6.8.17.1 Offset

Register	Offset
SGMII_LINK_TMR_H	13h

#### 33.6.8.17.2 Function

The SGMII Link Timer Register Upper contains bits 20:16 of the link timer value.

#### 33.6.8.17.3 Diagram



#### 33.6.8.17.4 Fields

Field	Function
0-10	Reserved
—	
11-15	Link Timer Upper
LINK_TMR_U	Link timer[20:16]

### 33.6.8.18 SGMII IF Mode (SGMII\_IF\_MODE)

#### 33.6.8.18.1 Offset

Register	Offset
SGMII_IF_MODE	14h

### 33.6.8.18.2 Function

The SGMII IF Mode Register contains control bits to set the interface mode.

### 33.6.8.18.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R												SGMII_DUPLEX	SGMII_SPEED		USE_SGMII_AN	SGMII_EN
W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.8.18.4 Fields

Field	Function
0-10 —	Reserved
11 SGMII_DUPLEX	SGMII Duplex SGMII Duplex Mode: Bit ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 0b - Full duplex enabled (default)
12-13 SGMII_SPEED	SGMII Speed SGMII Speed. When the PCS operates in SGMII mode (SGMII_EN set to 1) and is programmed not to be automatically configured (USE_SGMII_AN set to 0), sets the PCS speed of operation: Bits ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 00b - 10Mbps 01b - 100Mbps 10b - Gigabit 11b - Reserved
14 USE_SGMII_AN	Use SGMII Enable Use the SGMII Auto-Negotiation Results to Program the PCS Speed. When set to 0 (Reset Value), the PCS operation should be programmed with the register bit SGMII_SPEED and SGMII_DUPLEX. When set to 1, the PCS operation is automatically programmed with the Partner abilities advertised during Auto-Negotiation. Ignored when SGMII_EN is set to 0.
15 SGMII_EN	SGMII Enable SGMII Mode Enable. When set to '0' (Reset Value), the PCS operates in standard 1000Base-X Gigabit mode, when set to '1', the PCS operates in SGMII Mode

## 33.6.9 MDIO\_QSGMII register descriptions

The QSGMII MDIO register space is selected when the associated QSGMIIInCR1[MDEV\_PORT] matches the most-significant 3 bits of the Ethernet MAC PHY address (MDIO\_CTL[PHY\_ADDR]). The least significant 2 bits of the PHY address are used to select the specific QSGMII PCS port (0, 1, 2, or 3).

### 33.6.9.1 MDIO\_QSGMII memory map

MDIO\_QSGMII base address: 0h

Offset	Register	Width (in bits)	Access	Reset value
0h	QSGMII Control (QSGMII_CR)	16	RW	1140h
1h	QSGMII Status (QSGMII_SR)	16	RO	0009h
2h	QSGMII PHY Identifier Upper (QSGMII_PHY_ID_H)	16	RO	0083h
3h	QSGMII PHY Identifier Lower (QSGMII_PHY_ID_L)	16	RO	E400h
4h	QSGMII Device Ability for SGMII (QSGMII_DEV_ABIL_SGMII)	16	RW	01A0h
5h	QSGMII Partner Ability for SGMII (QSGMII_LP_DEV_ABIL_SGMII)	16	RO	0000h
6h	QSGMII AN Expansion (QSGMII_AN_EXP)	16	RO	0004h
7h	QSGMII Next Page Transmit (QSGMII_NP_TX)	16	RW	0000h
8h	QSGMII LP Next Page Receive (QSGMII_NP_RX)	16	RO	0000h
Fh	QSGMII Extended Status (QSGMII_XTND_STAT)	16	RU	0000h
10h	QSGMII Scratch (QSGMII_SCRATCH)	16	RW	0000h
11h	QSGMII Design Revision (QSGMII_REV)	16	RO	0001h
12h	QSGMII Link Timer Lower (QSGMII_LINK_TMR_L)	16	RW	12D0h
13h	QSGMII Link Timer Upper (QSGMII_LINK_TMR_H)	16	RW	0013h
14h	QSGMII IF Mode (QSGMII_IF_MODE)	16	RW	0000h

### 33.6.9.2 QSGMII Control (QSGMII\_CR)

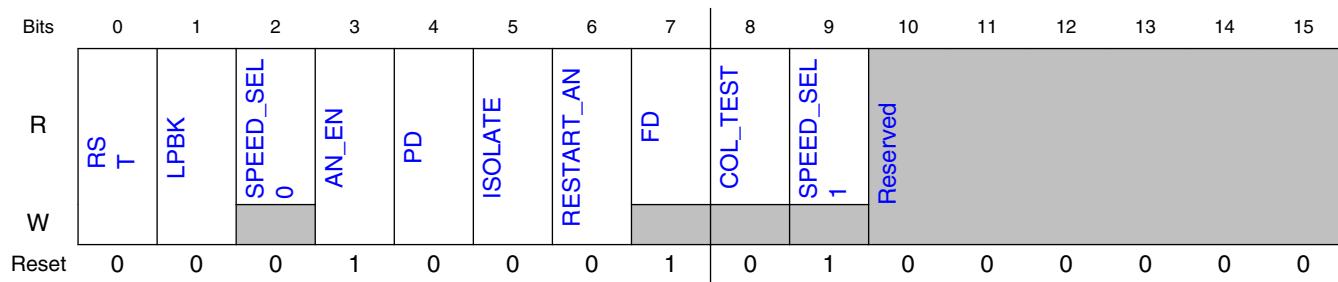
#### 33.6.9.2.1 Offset

Register	Offset
QSGMII_CR	0h

### 33.6.9.2.2 Function

The QSGMII Control Register contains control bits used to enable/disable functions in the PCS, and to initiate commands such as reset.

### 33.6.9.2.3 Diagram



### 33.6.9.2.4 Fields

Field	Function
0 RST	reset Self-Clearing Read / Write Reset Command Register. When set to 1, a synchronous reset pulse is generated which resets all the PCS state machines, the Comma detection function and the 8b/10b coder / decoder. Should be set to 0 (default) for normal operation.
1 LPBK	Loopback Loopback Command. 0b - Normal operation 1b - Reserved
2 SPEED_SEL0	Speed selection (LSB) Speed selection (LSB) All others reserved Note: SGMII speed is controlled with the IF Mode register MSB,LSB 1,0: 1000 Mb/s All others reserved
3 AN_EN	AN enable Auto-negotiation enable 0b - Disable auto-negotiation 1b - Enable auto-negotiation
4 PD	Power down Power down 0b - Normal operation 1b - Power down PCS. The PCS is held in internal soft reset until the bit is cleared again.
5	Isolate

Table continues on the next page...

Field	Function
ISOLATE	Isolate 0b - Normal operation 1b - Reserved
6 RESTART_AN	Restart AN Restart auto-negotiation This bit is self-clearing 0b - Normal operation 1b - Restart auto-negotiation
7 FD	Full Duplex Duplex mode. Read-only 0b - Reserved 1b - Full duplex
8 COL_TEST	Collision test Collision test. Read-only 0b - Disable COL signal test 1b - Reserved
9 SPEED_SEL1	Speed selection (MSB) Speed selection (MSB) See SPEED_SEL0
10-15 —	Reserved

### 33.6.9.3 QSGMII Status (QSGMII\_SR)

#### 33.6.9.3.1 Offset

Register	Offset
QSGMII_SR	1h

#### 33.6.9.3.2 Function

The QSGMII Status Register contains status bits on the operation of the PCS.

### 33.6.9.3.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ST_100BASET4	FD_100BASEX	HD_100BASEX	FD_10MBPS	HD_10MBPS	FD_100BASET2	HD_100BASET2	EXT_STAT	Reserved	AN_COMP	REM_FAULT	AN_ABIL	LINK_STAT	JAB_DET	EXT_CAP	
W									0	0	0	0	1	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	

### 33.6.9.3.4 Fields

Field	Function
0	100Base-T4
ST_100BASET4	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T4 operation
1	100Base-X Full Duplex
FD_100BASEX	Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
2	100Base-X Half Duplex
HD_100BASEX	Read Only bit set to 0 to indicate that the PCS does not support 100Base-X operation
3	10Mbps Full Duplex
FD_10MBPS	Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
4	10Mbps Half Duplex
HD_10MBPS	Read Only bit set to 0 to indicate that the PCS does not support 10Mbps operation
5	100Base-T2 Full Duplex
FD_100BASET2	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
6	100Base-T2 Half Duplex
HD_100BASET2	Read Only bit set to 0 to indicate that the PCS does not support 100Base-T2 operation.
7	Extended Status
EXT_STAT	Read Only bit always set to 0 to indicate that the PCS does not implement an extended status register.
8-9	Reserved
—	
10	AN Complete
AN_COMP	Auto-negotiation complete. Read Only Bit 0b - The Auto Negotiation process is not completed or Auto Negotiation is disabled 1b - The Auto Negotiation process is completed and that the Auto Negotiation control registers are valid.
11	remote Fault
REM_FAULT	Read Only Bit always set to 0. The PCS does not implement a PHY specific remote fault detection optional function.

Table continues on the next page...

Field	Function
12 AN_ABIL	AN Ability Auto Negotiation Ability. Read Only Bit set to „1. to indicate that the PCS supports Auto-Negotiation.
13 LINK_STAT	Link Status Link Status 0b - The link is not valid. If the link synchronization is lost a 0 is latched which is cleared only after a register read access 1b - This link is valid.
14 JAB_DET	Jabber Detect Read Only bit always set to 0, the Core does not support the optional Jabber detection function
15 EXT_CAP	Extended Capability Read Only bit set to „1. to indicate that the Core supports extended registers

### 33.6.9.4 QSGMII PHY Identifier Upper (QSGMII\_PHY\_ID\_H)

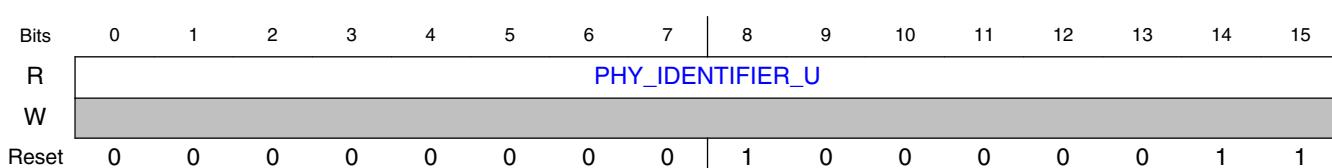
#### 33.6.9.4.1 Offset

Register	Offset
QSGMII_PHY_ID_H	2h

#### 33.6.9.4.2 Function

The QSGMII PHY Identifier Upper Register contains the upper half of the 32-bit PHY Identifier.

#### 33.6.9.4.3 Diagram



#### 33.6.9.4.4 Fields

Field	Function
0-15	PHY Identifier Upper PHY Identifier Upper:

## MDIO register spaces

Field	Function
PHY_IDENTIFIER_R_U	OUI[3:18]

### 33.6.9.5 QSGMII PHY Identifier Lower (QSGMII\_PHY\_ID\_L)

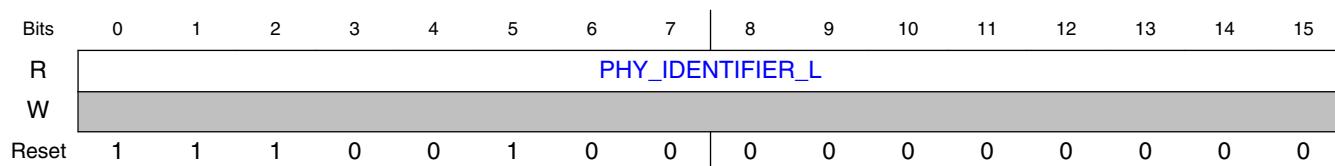
#### 33.6.9.5.1 Offset

Register	Offset
QSGMII_PHY_ID_L	3h

#### 33.6.9.5.2 Function

The QSGMII PHY Identifier Lower Register contains the lower half of the 32-bit PHY Identifier.

#### 33.6.9.5.3 Diagram



#### 33.6.9.5.4 Fields

Field	Function
0-15	PHY Identifier Lower
PHY_IDENTIFIER_R_L	PHY Identifier Lower: 15:10 OUI[19:24] 9:4 Manufacturer's Model Number 3:0 Revision Number

### 33.6.9.6 QSGMII Device Ability for SGMII (QSGMII\_DEV\_ABIL\_SGMII)

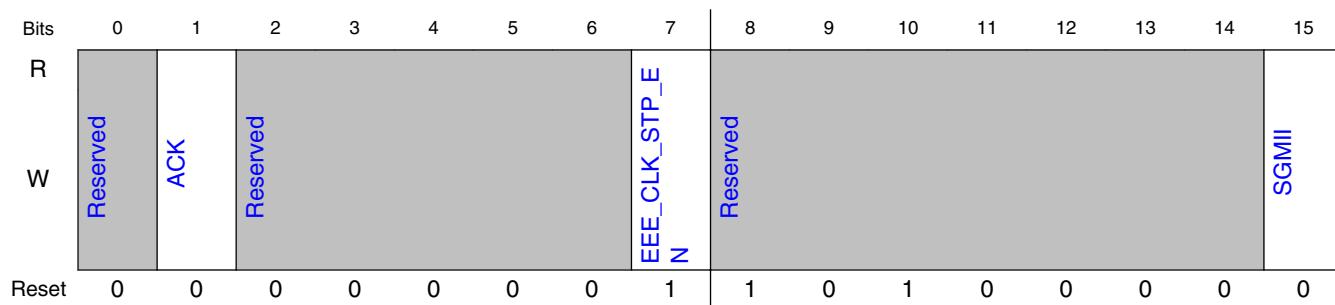
### 33.6.9.6.1 Offset

Register	Offset
QSGMII_DEV_ABIL_SGMII	4h

### 33.6.9.6.2 Function

The QSGMII Device Ability Register contains the control bits used to advertise the device abilities to the Link Partner during auto-negotiation for SGMII mode.

### 33.6.9.6.3 Diagram



### 33.6.9.6.4 Fields

Field	Function
0	Reserved
—	
1	Ack
ACK	Read only bit set to 1 when device has received three consecutive matching ability values from the link partner
2-6	Reserved. Should be set to 0
—	
7	EEE Clock Stop Enable
EEE_CLK_STP_EN	EEE Clock Stop Enable Should be set to 0 before SGMII Auto-Negotiation enable/restart, as this device does not support EEE clock stop. 0b - EEE Clock Stop Disabled 1b - EEE Clock Stop Enabled
8-14	Reserved. Should be set to 0
—	

Table continues on the next page...

## MDIO register spaces

Field	Function
15	SGMII
SGMII	SGMII mode. Should be set to 1.

### 33.6.9.7 QSGMII Partner Ability for SGMII (QSGMII\_LP\_DEV\_ABIL\_SGMII)

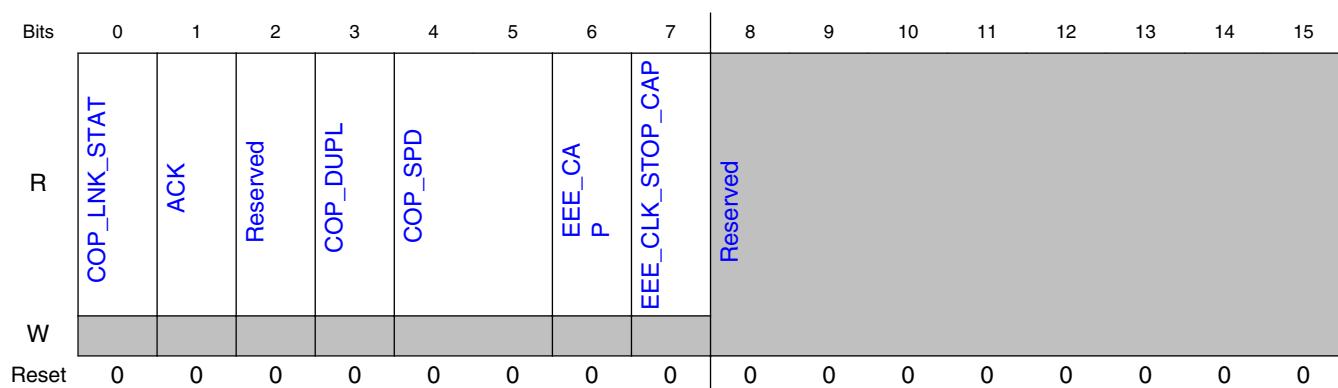
#### 33.6.9.7.1 Offset

Register	Offset
QSGMII_LP_DEV_ABIL_SGMII	5h

#### 33.6.9.7.2 Function

The QSGMII Partner Ability Register contains the capability status of the SGMII link partner.

#### 33.6.9.7.3 Diagram



#### 33.6.9.7.4 Fields

Field	Function
0 COP_LNK_STA T	copper Link Status Read only bit, used by the SGMII PHY to advertise the Link Partner Copper status: 0b - Copper interface link is down

*Table continues on the next page...*

Field	Function
	1b - Copper interface link is up
1 ACK	Ack Read only bit set to 1. when the Link Partner Copper Interface advertises that it has received three consecutive matching ability values from the device
2 —	Always 0
3 COP_DUPL	Copper Duplex Read only bit, used by the SGMII PHY to advertise the Link Copper duplex capability: 0b - Copper Interface resolved to Half-Duplex 1b - Copper Interface resolved to Full-Duplex
4-5 COP_SPD	Copper Speed Read only bits, used by the SGMII PHY to advertise the Link Partner Copper interface speed 00b - Copper Interface Speed is 10Mbps 01b - Copper Interface Speed is 100Mbps 10b - Copper Interface Speed is Gigabit 11b - Reserved
6 EEE_CAP	EEE Capability EEE Capability 0b - EEE not supported 1b - EEE supported
7 EEE_CLK_STO P_CAP	EEE Clock Stop Capability EEE Clock Stop Capability 0b - EEE Clock Stop not supported 1b - EEE Clock Stop supported
8-15 —	Reserved

### 33.6.9.8 QSGMII AN Expansion (QSGMII\_AN\_EXP)

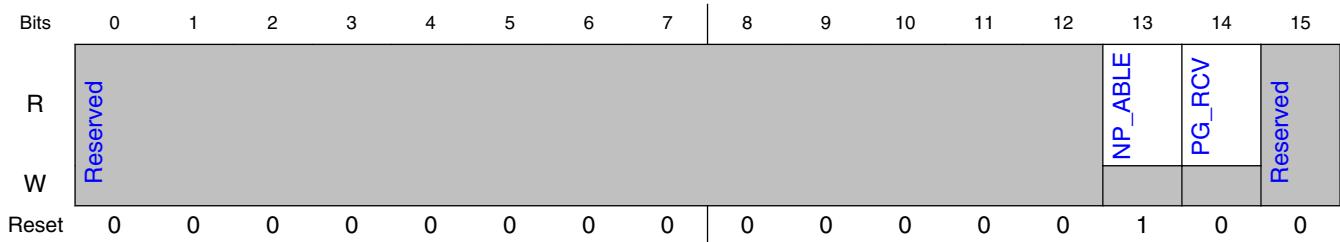
#### 33.6.9.8.1 Offset

Register	Offset
QSGMII_AN_EXP	6h

#### 33.6.9.8.2 Function

The QSGMII AN Expansion Register contains status bits indicating the PCS next page auto-negotiation capability and status.

### 33.6.9.8.3 Diagram



### 33.6.9.8.4 Fields

Field	Function
0-12 —	Reserved
13 NP_ABLE	Next Page Able Read Only bit set to 1 to indicate the PCS does support the Next Page function
14 PG_RCV	Page received Set to 1 to indicate that a new page has been received with new partner ability available in the PCS register PARTNER_ABILITY. The bit is set to 0 (Reset value) when the system management agent performs a read access.
15 —	Reserved

## 33.6.9.9 QSGMII Next Page Transmit (QSGMII\_NP\_TX)

### 33.6.9.9.1 Offset

Register	Offset
QSGMII_NP_TX	7h

### 33.6.9.9.2 Function

The QSGMII NP TX Register contains next page data to transfer to the remote device. Writing to this register initiates a next page exchange (sets mr\_np\_loaded variable).

### 33.6.9.9.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
R	NP	ACK	MP	ACK2	TOGGLE	DATA											
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 33.6.9.9.4 Fields

Field	Function
0 NP	Next Page Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1 ACK	Ack Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2 MP	Message page Message page (1) or unformatted page (0) format.
3 ACK2	Ack Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4 TOGGLE	Toggle The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15 DATA	Data 11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

## 33.6.9.10 QSGMII LP Next Page Receive (QSGMII\_NP\_RX)

### 33.6.9.10.1 Offset

Register	Offset
QSGMII_NP_RX	8h

### 33.6.9.10.2 Function

The QSGMII NP RX Register contains the next page data received from the remote device during the latest next page exchange. The value is overridden with every new next page. Next page transfers are controlled by the application.

### 33.6.9.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	NP	ACK	MP	ACK2	TOGGLE	DATA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.9.10.4 Fields

Field	Function
0 NP	Next Pge Next page indication. Indicates if additional next pages will follow (1) or this is the last page (0).
1 ACK	Ack Acknowledge bit used by the autoneg function during message transfer. It has no meaning to the application and should be written with 0 always and ignored on read.
2 MP	Message Page Message page (1) or unformatted page (0) format.
3 ACK2	Ack Application level acknowledge to indicate acceptance of a message. Use is defined by application layer.
4 TOGGLE	Toggle The toggle bit is used by the auto-negotiation function to keep track of message exchange. It has no meaning to the application and should be written with 0 always and ignored on read.
5-15 DATA	Data 11 bits of data which can either be a message-id (if MP bit 13=1) or unformatted data (if MP bit 13=0). See IEEE 802.3 Annex 28C for message page encodings

## 33.6.9.11 QSGMII Extended Status (QSGMII\_XTND\_STAT)

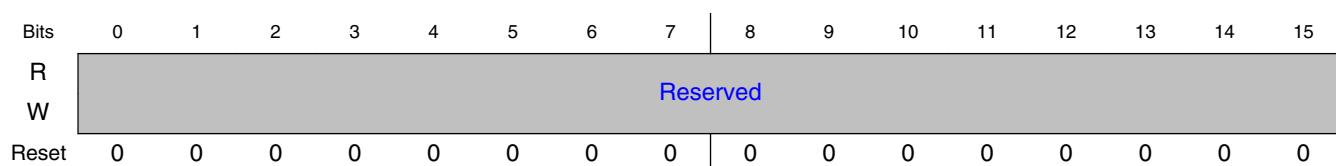
### 33.6.9.11.1 Offset

Register	Offset
QSGMII_XTND_STAT	Fh

### 33.6.9.11.2 Function

The QSGMII Extended Status Register is reserved, as this device does not implement the optional extended status registers.

### 33.6.9.11.3 Diagram



### 33.6.9.11.4 Fields

Field	Function
0-15	Reserved. Always 0
—	

## 33.6.9.12 QSGMII Scratch (QSGMII\_SCRATCH)

### 33.6.9.12.1 Offset

Register	Offset
QSGMII_SCRATCH	10h

### 33.6.9.12.2 Function

The QSGMII Scratch Register provides a memory location available to test register read and write operations.

### 33.6.9.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									SCRATCH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 33.6.9.12.4 Fields

Field	Function
0-15	Scratch
SCRATCH	Scratch field.

## 33.6.9.13 QSGMII Design Revision (QSGMII\_REV)

### 33.6.9.13.1 Offset

Register	Offset
QSGMII_REV	11h

### 33.6.9.13.2 Function

The QSGMII Revision Register contains revision information for the PCS.

### 33.6.9.13.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									Reserved			IP_MJ		IP_MN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

### 33.6.9.13.4 Fields

Field	Function
0-7	Reserved

Table continues on the next page...

Field	Function
—	
8-11	Major revision
IP_MJ	Major revision
12-15	Minor revision
IP_MN	Minor revision

### 33.6.9.14 QSGMII Link Timer Lower (QSGMII\_LINK\_TMR\_L)

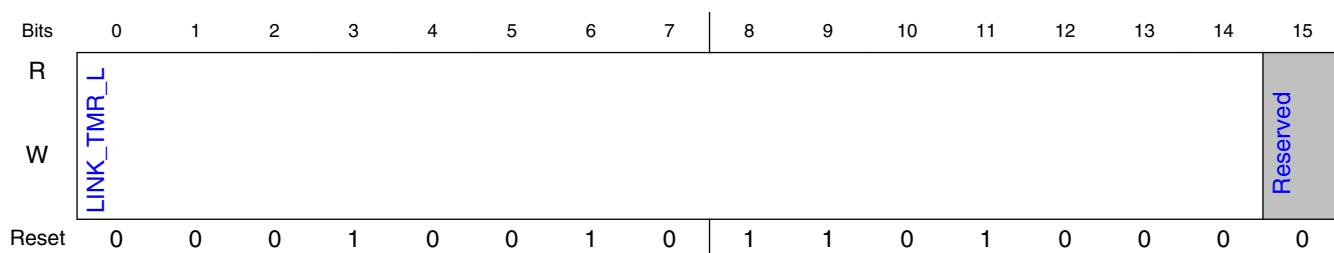
#### 33.6.9.14.1 Offset

Register	Offset
QSGMII_LINK_TMR_L	12h

#### 33.6.9.14.2 Function

The QSGMII Link Timer Register Lower contains bits 15:1 of the link timer value.

#### 33.6.9.14.3 Diagram



#### 33.6.9.14.4 Fields

Field	Function
0-14 LINK_TMR_L	Link Timer Lower Link timer[15:1]
15 —	Reserved

### 33.6.9.15 QSGMII Link Timer Upper (QSGMII\_LINK\_TMR\_H)

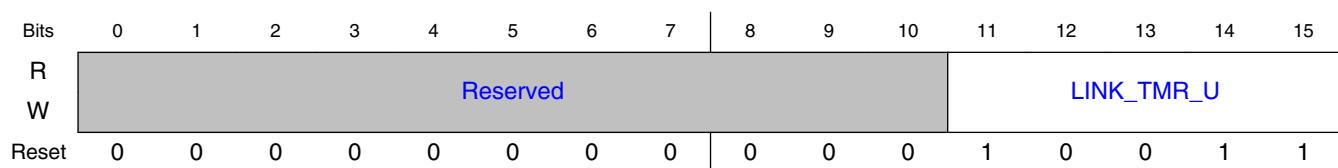
#### 33.6.9.15.1 Offset

Register	Offset
QSGMII_LINK_TMR_H	13h

#### 33.6.9.15.2 Function

The QSGMII Link Timer Register Upper contains bits 20:16 of the link timer value.

#### 33.6.9.15.3 Diagram



#### 33.6.9.15.4 Fields

Field	Function
0-10	Reserved
—	
11-15	Link Timer Upper
LINK_TMR_U	Link timer[20:16]

### 33.6.9.16 QSGMII IF Mode (QSGMII\_IF\_MODE)

#### 33.6.9.16.1 Offset

Register	Offset
QSGMII_IF_MODE	14h

### 33.6.9.16.2 Function

The QSGMII IF Mode Register contains control bits to set the interface mode.

### 33.6.9.16.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R												SGMII_DUPLEX	SGMII_SPEED			
W	Reserved											0	0	0	0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 33.6.9.16.4 Fields

Field	Function
0-10 —	Reserved
11 SGMII_DUPLEX	SGMII Duplex SGMII Duplex Mode: Bit ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 0b - Full duplex enabled (default)
12-13 SGMII_SPEED	SGMII Speed SGMII Speed. When the PCS operates in SGMII mode (SGMII_EN set to 1) and is programmed not to be automatically configured (USE_SGMII_AN set to 0), sets the PCS speed of operation: Bits ignored when SGMII_EN is set to 0 or USE_SGMII_AN is set to 1. 00b - 10Mbps 01b - 100Mbps 10b - Gigabit 11b - Reserved
14 USE_SGMII_AN	Use SGMII AN Use the SGMII Auto-Negotiation Results to Program the PCS Speed. When set to 0 (Reset Value), the PCS operation should be programmed with the register bit SGMII_SPEED and SGMII_DUPLEX. When set to 1, the PCS operation is automatically programmed with the Partner abilities advertised during Auto-Negotiation. Ignored when SGMII_EN is set to 0.
15 SGMII_EN	SGMII Enable SGMII Mode Enable. When set to '0' (Reset Value), the PCS operates in standard 1000Base-X Gigabit mode, when set to '1', the PCS operates in SGMII Mode

## 33.7 Initialization/Application Information

### 33.7.1 Initialization

All initialization necessary for the SerDes module to start operation is done automatically based on RCW, with the exception of the following protocols:

#### 33.7.1.1 1G SGMII

After initializing the MAC, and prior to initiating any SGMII traffic, perform the following SGMII protocol initialization:

1. SGMII IF Mode register:
  - SGMII\_EN=1
  - USE\_SGMII\_AN=1 if SGMII auto-negotiation is desired
  - SGMII\_SPEED (if USE\_SGMII\_AN=0)
  - SGMII\_DUPLEX (if USE\_SGMII\_AN=0)
2. SGMII Device Ability Register:
  - EEE\_CLK\_STP\_EN=0
  - SGMII=1

To enable SGMII auto-negotiation, perform the following sequence:

1. SGMII Link Timer Register Upper:
  - LINK\_TMR\_U=0h03
2. SGMII Link Timer Register Lower:
  - LINK\_TMR\_L=0h06A0
3. SGMII Control Register:
  - AN\_EN=1
  - RESTART\_AN=1

Note: Half duplex is not supported.

#### 33.7.1.2 2.5G SGMII

All default SerDes settings are for 1G SGMII rather than 2.5G SGMII. The following settings need to be updated for 2.5G SGMII, following the procedure in [Lane Reset and Reconfiguration](#), prior to initializing the SGMII PCS:

- LNmGCR1[REIDL\_TH]
- LNmGCR1[REIDL\_EX\_SEL]
- LNmGCR1[REIDL\_ET\_MSB]

- LNmGCR1[ISLEW\_RCTL]
- LNmGCR1[OSLEW\_RCTL]
- LNmRECR0[GK2OVD]
- LNmRECR0[GK3OVD]
- LNmRECR0[GK2OVD\_EN]
- LNmRECR0[GK3OVD\_EN]
- LNmTECR0[TEQ\_TYPE]
- LNmTECR0[RATIO\_PST1Q]
- LNmTECR0[AMP\_RED]

See [General Control Register 1 - Lane a \(LNAGCR1 - LNDGCR1\)](#), [Receive Equalization Control Register 0 - Lane a \(LNARECR0 - LNDRECR0\)](#) and [Transmit Equalization Control Register 0 - Lane a \(LNATECR0 - LNDTECR0\)](#) for details.

The PCS initialization sequence for 2.5G SGMII is the same as for 1G SGMII with auto-negotiation disabled (see [1G SGMII](#)). Auto-negotiation is not supported for 2.5G SGMII.

### 33.7.1.3 1000Base-KX

All default SerDes settings are for 1G SGMII rather than 1000Base-KX. The following settings need to be updated for 1000Base-KX, following the procedure in [Lane Reset and Reconfiguration](#), prior to performing 1000Base-KX auto-negotiation:

- LNmGCR1[REIDL\_TH]
- LNmGCR1[REIDL\_EX\_SEL]
- LNmGCR1[REIDL\_ET\_MSB]
- LNmTECR0[AMP\_RED]

See [General Control Register 1 - Lane a \(LNAGCR1 - LNDGCR1\)](#) and [Transmit Equalization Control Register 0 - Lane a \(LNATECR0 - LNDTECR0\)](#) for details.

After initializing the MAC and SERDES lane(s), and prior to initiating any 1000Base-KX traffic, perform the following 1000Base-KX protocol initialization:

1. Enable the 1000Base-KX AN reference clock by setting PLLnCR0[DLYDIV\_SEL]=01, for the PLLn that is the clock source for the SGMII PCS (see TPLL\_LES in [General Control Register 0 - Lane a \(LNAGCR0 - LNDG CR0\)](#)).
2. Enable the 1000Base-KX AN module by setting PCCR8[SGMIIp\_KX]=1, for each SGMIIp that will run in 1000Base-KX rather than SGMII mode.
3. Initialize SGMII IF Mode register to 0x0008:
  - SGMII\_EN=0

- USE\_SGMII\_AN=0
  - SGMII\_SPEED=10
4. Initialize 1000Base-KX (Clause 45) AN Advertisement Register 1:
    - TX\_NONCE=unique value per device
  5. Read 1000Base-KX (Clause 45) AN Status Register to clear any previous status
  6. Initialize 1000Base-KX (Clause 45) AN Control Register to 0x1200:
    - AN\_ENAB=1
    - RST\_AN=1

### **33.7.1.4 QSGMII**

After initializing the MACs, and prior to initiating any QSGMII traffic, perform the following per-port SGMII protocol initialization in each corresponding register in the QSGMII PCS (4 per PCS):

1. QSGMII IF Mode register:
  - SGMII\_EN=1
  - USE\_SGMII\_AN=1 if SGMII auto-negotiation is desired
  - SGMII\_SPEED (if USE\_SGMII\_AN=0)
  - SGMII\_DUPLEX (if USE\_SGMII\_AN=0)
2. QSGMII Device Ability Register:
  - SGMII=1
3. QSGMII Link Timer Register Upper:
  - SGMII: LINK\_TMR\_U=0h03
4. QSGMII Link Timer Register Lower:
  - SGMII: LINK\_TMR\_L=0h06A0
5. QSGMII Control Register:
  - RESTART\_EN=1

To initialize QSGMII with auto-negotiation disabled, instead of steps 2-4 above, do:

6. QSGMII Control Register:
  - AN\_EN=0

### **33.7.1.5 XFI**

All default SerDes settings are for XFI, and only settings that differ for are listed separately. There is no additional initialization required for XFI operation.

## 33.7.2 Unused Lanes

Unused lanes should be powered down to save power and avoid noise on adjacent lanes.

Power down the Rx portion of the lanes by setting LNmGCR0[RX\_PD]=1 and LNmGCR0[RRST\_B]=0.

Power down the Tx portion of the lanes by setting LNmGCR0[TX\_PD]=1 and LNmGCR0[TRST\_B]=0.

The Rx and Tx portions of the lanes may be independently powered down for uni-directional lanes or links.

## 33.7.3 Soft Reset and Reconfiguring Procedures

### 33.7.3.1 Lane Reset and Reconfiguration

To reconfigure a lane (change settings including clock divider or PLL select), perform the following sequence:

1. Put the lane(s) into reset by setting LNmGCR0[TRST\_B]=0 and LNmGCR0[RRST\_B]=0.
2. Wait at least 50 ns
3. Change the desired per-lane settings
4. Wait at least 120 ns
5. Take the lane(s) out of reset by setting LNmGCR0[TRST\_B]=1 and LNmGCR0[RRST\_B]=1

Note that if the lanes being reconfigured are grouped for multi-lane or synchronous mode, then the master source clock lane for the group must have TRST\_B set to 1 after all the other lanes in the group. The master source clock lane of the group is indicated by LNmGCR0[1STLANE]=1. All lanes p<m (if LNmGCR1[TRSTDIR]=0) or p>m (if LNmGCR1[TRSTDIR]=1) of the master source clock lane until the next lane with LNmGCR0[1STLANE]=1, or the end of the SerDes, are grouped with the master source clock lane. All grouped lanes must have the same setting of TRSTDIR.

It is recommended to disable any controller connected to a lane being reconfigured prior to starting the reconfiguration sequence.

### 33.7.3.2 Lane Enable After Powerdown

To enable a previously powered down lane, set LNmGCR0[nX\_PD]=0 (n=R or T), wait 15 us, then set LNmGCR0[nRST]=1.

Note that if a Tx lane m with LNmGCR0[1STLANE]=1 (master Tx clock lane) is powered down or reset, all Tx lanes p<m (if LNmGCR1[TRSTDIR]=0), or all Tx lanes p>m (if LNmGCR1[TRSTDIR]=1), cannot be used.

### 33.7.3.3 PLL Reset and Reconfiguration

To reconfigure a PLL, perform the following sequence:

1. Disable all lanes using the PLL to be reconfigured by setting  
PLL<sub>n</sub>RSTCTL[SDRST\_B]=0
2. Wait at least 50 ns
3. Disable the PLL by setting PLL<sub>n</sub>RSTCTL[SDEN]=0 and  
PLL<sub>n</sub>RSTCTL[PLLRST\_B]=0
4. Wait at least 100 ns
5. Change the desired per-PLL settings (VCO frequency, refclk frequency, etc.).
6. Reset the PLL by setting PLL<sub>n</sub>RSTCTL[RSTREQ]=1
7. Set PLL<sub>n</sub>RSTCTL[SDEN]=1, PLL<sub>n</sub>RSTCTL[PLLRST\_B]=1, and  
PLL<sub>n</sub>RSTCTL[SDRST\_B]=1
  - Note this step is not to be combined with setting RSTREQ=1

Note: lane reconfiguration may also be performed while the PLL is disabled by following the first three steps of the lane reset sequence in [Lane Reset and Reconfiguration](#) after changing the per-PLL settings, before setting RSTREQ=1, and the last step of the lane reset sequence after the last step of the above PLL reset sequence.

It is recommended to disable any controller connected to a lane selecting the PLL being reconfigured prior to starting the reconfiguration sequence. In the case of PCI Express, the controllers must be disabled to prevent auto-negotiation to 8Gbaud, which can include autonomous PLL reset and reconfiguration.

### 33.7.4 Quiesce Sequences for System Sleep

To quiesce the SerDes module in preparation for System Sleep, the user must first quiesce all controllers and disable transmission/reception of packets.

# Chapter 34

## Serial Peripheral Interface (SPI)

### 34.1 The SPI module as implemented on the chip

This section provides details about how the SPI module is implemented on the chip.

#### 34.1.1 LS1043A SPI signals

The following table lists the SoC signal names and their corresponding SPI module signal names used in this chapter:

**Table 34-1. LS1043A SPI signals**

LS1043A signal name	SPI module signal
SPI_CS_B[0]	PCS0
SPI_CS_B[1:3]	PCS[1:3]
Unused	PCS4/PCS5
SPI_CLK	SCK
SPI_MISO	SIN
SPI_MOSI	SOUT

#### 34.1.2 LS1043A SPI module integration

The following table describes the SPI module integration into this chip:

**Table 34-2. SPI module integration**

Module	Module Base address
SPI1	210_0000

The remainder of this chapter refers to a single SPI module. Notes are included to indicate variations for multiple instantiations.

### **34.1.3 LS1043A SPI module special consideration**

The SPI module implements the following parameter settings in the chip:

**Table 34-3. LS1043A SPI parameter settings**

SPI parameters	LS1043A parameter value
NUM_CTAR_PP	2
TX_FIFO_DEPTH_PP	16
RX_FIFO_DEPTH_PP	16
No. of SPI	1
SP Chip Selects	4
Stop mode support	Yes. Refers to the LPM20 low power mode of the chip.
Debug mode support	No

## **34.2 Introduction**

The serial peripheral interface (SPI) module provides a synchronous serial bus for communication between a chip and an external peripheral device.

### **34.2.1 Block Diagram**

The block diagram of this module is as follows:

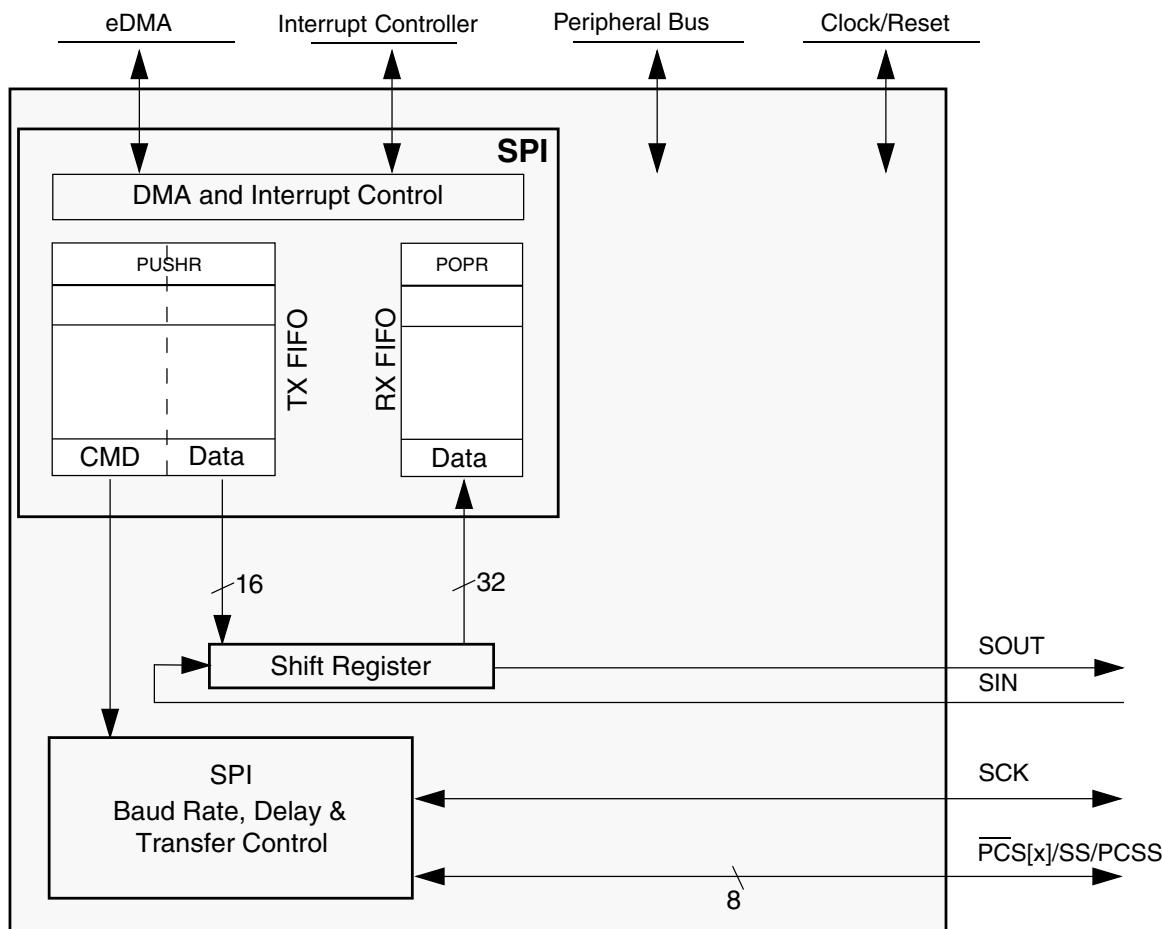


Figure 34-1. SPI Block Diagram

### 34.2.2 Features

The module supports the following features:

- Full-duplex, three-wire synchronous transfers
- Master mode
- Buffered transmit operation using the transmit first in first out (TX FIFO) with depth of 16 entries
- Support for 8/16-bit accesses to the PUSH TX FIFO Register Data Field
- Buffered receive operation using the receive FIFO (RX FIFO) with depth of 16 entries
- Asynchronous clocking scheme for Register and Protocol Interfaces
- TX and RX FIFOs can be disabled individually for low-latency updates to SPI queues

- Visibility into TX and RX FIFOs for ease of debugging
- Programmable transfer attributes on a per-frame basis:
  - four transfer attribute registers
  - four extended transfer attribute registers
  - Serial clock (SCK) with programmable polarity and phase
  - Various programmable delays
  - Programmable serial frame size: 4 to 32
    - SPI frames longer than 32 bits can be supported using the continuous selection format.
  - Continuously held chip select capability
  - Parity control
- 4 peripheral chip selects (PCSes), expandable to 16 with external demultiplexer
- Deglitching support for up to 8 peripheral chip selects (PCSes) with external demultiplexer
- DMA support for adding entries to TX FIFO and removing entries from RX FIFO:
  - TX FIFO is not full (TFFF)
  - RX FIFO is not empty (RFDF)
  - CMD FIFO is not full (CMDFFF)
- Interrupt conditions:
  - End of Queue reached (EOQF)
  - TX FIFO is not full (TFFF)
  - CMD FIFO is not full (CMDFFF)
  - Transfer of current frame complete (TCF)
  - Transfers due from current command frame complete (CMDTCF)
  - Attempt to transmit with an empty Transmit FIFO (TFUF)
  - RX FIFO is not empty (RFDF)
  - Frame received while Receive FIFO is full (RFOF)

- SPI Parity Error (SPEF)
- Data present in TX FIFO while CMD FIFO is empty (TFIWF)
- Global interrupt request line
- Power-saving architectural features:
  - Support for Stop mode
  - Support for Doze mode

### 34.2.3 Interface configurations

#### 34.2.3.1 SPI configuration

The Serial Peripheral Interface SPI configuration allows the module to send and receive serial data. This configuration allows the module to operate as a basic SPI block with internal FIFOs supporting external queue operation. Transmitted data and received data reside in separate FIFOs. The host CPU or a DMA controller read the received data from the Receive FIFO and write transmit data to the Transmit FIFO.

For queued operations, the SPI queues can reside in system RAM, external to the module. Data transfers between the queues and the module FIFOs are accomplished by a DMA controller or host CPU. The following figure shows a system example with DMA, SPI, and external queues in system RAM.

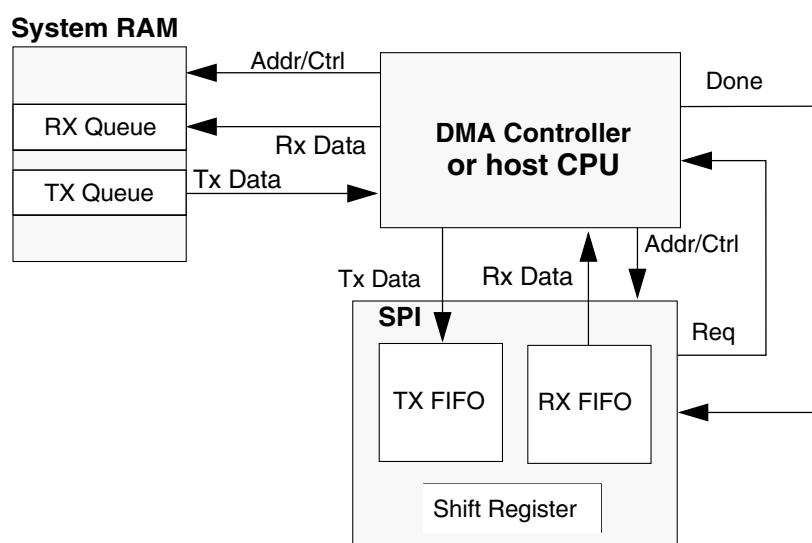


Figure 34-2. SPI with queues and DMA

## 34.2.4 Modes of Operation

The module supports the following modes of operation that can be divided into two categories:

- Module-specific modes:
  - Master mode
  - Module Disable mode
- Chip-specific modes:
  - External Stop mode

The module enters module-specific modes when the host writes a module register. The chip-specific modes are controlled by signals external to the module. The chip-specific modes are modes that a chip may enter in parallel to the block-specific modes.

### 34.2.4.1 Master Mode

Master mode allows the module to initiate and control serial communication. In this mode, these signals are controlled by the module and configured as outputs:

- SCK
- SOUT
- PCS[ $x$ ]

### 34.2.4.2 Module Disable Mode

The Module Disable mode can be used for chip power management. The clock to the non-memory mapped logic in the module can be stopped while in the Module Disable mode.

### 34.2.4.3 External Stop Mode

External Stop mode is used for chip power management. The module supports the Peripheral Bus Stop mode mechanism. When a request is made to enter External Stop mode, it acknowledges the request and completes the transfer that is in progress. When the module reaches the frame boundary, it signals that the protocol clock to the module may be shut off.

**NOTE**

In master mode, if a serial transfer is in progress, then this module waits until it reaches the frame boundary before it is ready to have its clocks shut off. In slave mode, this module waits until its chip select input signal goes high before it is ready to have its clocks shut off. It should be noted that the CS0 pad must be correctly configured to allow it to return to high when no transfer is occurring. If the pad is tied low then this could prevent the module from being correctly disabled.

### 34.3 Module signal descriptions

This table describes the signals on the boundary of the module that may connect off chip (in alphabetical order).

**Table 34-4. Module signal descriptions**

Signal	Master mode	Slave mode	I/O
PCS0	Peripheral Chip Select 0 (O)		I/O
PCS[1:3]	Peripheral Chip Selects 1–3	(Unused)	O
SCK	Serial Clock (O)	Serial Clock (I)	I/O
SIN	Serial Data In	Serial Data In	I
SOUT	Serial Data Out	Serial Data Out	O

#### 34.3.1 PCS0—Peripheral Chip Select

Master mode: Peripheral Chip Select 0 (O)—Selects an SPI slave to receive data transmitted from the module.

**NOTE**

Do not tie the SPI slave select pin to ground. Otherwise, SPI cannot function properly.

#### 34.3.2 PCS1–PCS3—Peripheral Chip Selects 1–3

Master mode: Peripheral Chip Selects 1–3 (O)—Select an SPI slave to receive data transmitted by the module.

### 34.3.3 SCK—Serial Clock

Master mode: Serial Clock (O)—Supplies a clock signal from the module to SPI slaves.

### 34.3.4 SIN—Serial Input

Master mode: Serial Input (I)—Receives serial data.

### 34.3.5 SOUT—Serial Output

Master mode: Serial Output (O)—Transmits serial data.

## 34.4 Memory Map/Register Definition

Register accesses to memory addresses that are reserved or undefined result in a transfer error. Any Write access to the POPR and RXFRn also results in a transfer error.

#### NOTE

$f_P$  and  $f_{sys}$  are used interchangeably in this chapter.

#### NOTE

While the module is in the running state, do not write to these registers:

- CTAREn

SPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
210_0000	Module Configuration Register (SPI_MCR)	32	R/W	0000_0001h	<a href="#">34.4.1/2165</a>
210_0008	Transfer Count Register (SPI_TCR)	32	R/W	0000_0000h	<a href="#">34.4.2/2169</a>
210_000C	Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR0)	32	R/W	7800_0000h	<a href="#">34.4.3/2169</a>
210_0010	Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR1)	32	R/W	7800_0000h	<a href="#">34.4.3/2169</a>
210_0014	Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR2)	32	R/W	7800_0000h	<a href="#">34.4.3/2169</a>
210_0018	Clock and Transfer Attributes Register (In Master Mode) (SPI_CTAR3)	32	R/W	7800_0000h	<a href="#">34.4.3/2169</a>

Table continues on the next page...

**SPI memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
210_002C	Status Register (SPI_SR)	32	R/W	0201_0000h	<a href="#">34.4.4/2174</a>
210_0030	DMA/Interrupt Request Select and Enable Register (SPI_RSER)	32	R/W	0000_0000h	<a href="#">34.4.5/2177</a>
210_0034	PUSH TX FIFO Register In Master Mode (SPI_PUSHR)	32	R/W	0000_0000h	<a href="#">34.4.6/2180</a>
210_0038	POP RX FIFO Register (SPI_POPR)	32	R	0000_0000h	<a href="#">34.4.7/2182</a>
210_003C	Transmit FIFO Registers (SPI_TXFR0)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0040	Transmit FIFO Registers (SPI_TXFR1)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0044	Transmit FIFO Registers (SPI_TXFR2)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0048	Transmit FIFO Registers (SPI_TXFR3)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_004C	Transmit FIFO Registers (SPI_TXFR4)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0050	Transmit FIFO Registers (SPI_TXFR5)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0054	Transmit FIFO Registers (SPI_TXFR6)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0058	Transmit FIFO Registers (SPI_TXFR7)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_005C	Transmit FIFO Registers (SPI_TXFR8)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0060	Transmit FIFO Registers (SPI_TXFR9)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0064	Transmit FIFO Registers (SPI_TXFR10)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0068	Transmit FIFO Registers (SPI_TXFR11)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_006C	Transmit FIFO Registers (SPI_TXFR12)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0070	Transmit FIFO Registers (SPI_TXFR13)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0074	Transmit FIFO Registers (SPI_TXFR14)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_0078	Transmit FIFO Registers (SPI_TXFR15)	32	R	0000_0000h	<a href="#">34.4.8/2183</a>
210_007C	Receive FIFO Registers (SPI_RXFR0)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0080	Receive FIFO Registers (SPI_RXFR1)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0084	Receive FIFO Registers (SPI_RXFR2)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0088	Receive FIFO Registers (SPI_RXFR3)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_008C	Receive FIFO Registers (SPI_RXFR4)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0090	Receive FIFO Registers (SPI_RXFR5)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0094	Receive FIFO Registers (SPI_RXFR6)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_0098	Receive FIFO Registers (SPI_RXFR7)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_009C	Receive FIFO Registers (SPI_RXFR8)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00A0	Receive FIFO Registers (SPI_RXFR9)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00A4	Receive FIFO Registers (SPI_RXFR10)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00A8	Receive FIFO Registers (SPI_RXFR11)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00AC	Receive FIFO Registers (SPI_RXFR12)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00B0	Receive FIFO Registers (SPI_RXFR13)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00B4	Receive FIFO Registers (SPI_RXFR14)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>
210_00B8	Receive FIFO Registers (SPI_RXFR15)	32	R	0000_0000h	<a href="#">34.4.9/2184</a>

Table continues on the next page...

**SPI memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
210_011C	Clock and Transfer Attributes Register Extended (SPI_CTARE0)	32	R/W	0000_0001h	<a href="#">34.4.10/ 2184</a>
210_0120	Clock and Transfer Attributes Register Extended (SPI_CTARE1)	32	R/W	0000_0001h	<a href="#">34.4.10/ 2184</a>
210_0124	Clock and Transfer Attributes Register Extended (SPI_CTARE2)	32	R/W	0000_0001h	<a href="#">34.4.10/ 2184</a>
210_0128	Clock and Transfer Attributes Register Extended (SPI_CTARE3)	32	R/W	0000_0001h	<a href="#">34.4.10/ 2184</a>
210_013C	Status Register Extended (SPI_SREX)	32	R	0000_0000h	<a href="#">34.4.11/ 2186</a>

### 34.4.1 Module Configuration Register (SPI\_MCR)

Contains bits to configure various attributes associated with the module operations. The HALT and MDIS bits can be changed at any time, but the effect takes place only on the next frame boundary. Only the HALT and MDIS bits in the MCR can be changed, while the module is in the Running state.

Address: 210\_0000h base + 0h offset = 210\_0000h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
MSTR	0	0		DCONF	0	Reserved	Reserved	ROOE		Reserved				PCSIS		
W			CONT_SCKE													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## Memory Map/Register Definition

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R					0	0	0				0					
DOZE		MDIS	DIS_TXF	DIS_RXF									XSPI	FCPCS	PES	HALT
W					CLR_TXF	CLR_RXF										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

### SPI\_MCR field descriptions

Field	Description
0 MSTR	Master/Slave Mode Select Enables either Master mode (if supported) or Slave mode (if supported) operation. 0 Reserved 1 Enables Master mode
1 CONT_SCKE	Continuous SCK Enable Enables the Serial Communication Clock (SCK) to run continuously. 0 Continuous SCK disabled. 1 Continuous SCK enabled.
2–3 DCONF	SPI Configuration. Selects among the different configurations of the module. 00 SPI 01 Reserved 10 Reserved 11 Reserved
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved.
6 Reserved	This field is reserved.
7 ROOE	Receive FIFO Overflow Overwrite Enable

Table continues on the next page...

**SPI\_MCR field descriptions (continued)**

Field	Description										
	<p>In the RX FIFO overflow condition, configures the module to ignore the incoming serial data or overwrite existing data. If the RX FIFO is full and new data is received, the data from the transfer, generating the overflow, is ignored or shifted into the shift register.</p> <p>0 Incoming data is ignored. 1 Incoming data is shifted into the shift register.</p>										
8–11 Reserved	<p>Always write the reset value to this field.</p> <p>This field is reserved.</p>										
12–15 PCSIS	<p>Peripheral Chip Select x Inactive State</p> <p>Determines the inactive state of PCSx. Refer to the chip-specific SPI information for the number of PCS signals used in this chip.</p>										
	<p><b>Table 34-5. PCSIS signal and bit position mapping</b></p> <table border="1"> <thead> <tr> <th>Signal</th><th>Bit position</th></tr> </thead> <tbody> <tr> <td>PCSIS0</td><td>15</td></tr> <tr> <td>PCSIS1</td><td>14</td></tr> <tr> <td>PCSIS2</td><td>13</td></tr> <tr> <td>PCSIS3</td><td>12</td></tr> </tbody> </table> <p><b>NOTE:</b> The effect of this bit only takes place when module is enabled. Ensure that this bit is configured correctly before enabling the SPI interface.</p> <p>0 The inactive state of PCSx is low. 1 The inactive state of PCSx is high.</p>	Signal	Bit position	PCSIS0	15	PCSIS1	14	PCSIS2	13	PCSIS3	12
Signal	Bit position										
PCSIS0	15										
PCSIS1	14										
PCSIS2	13										
PCSIS3	12										
16 DOZE	<p>Doze Enable</p> <p>Provides support for an externally controlled Doze mode power-saving mechanism.</p> <p>0 Doze mode has no effect on the module. 1 Doze mode disables the module.</p>										
17 MDIS	<p>Module Disable</p> <p>Allows the clock to be stopped to the non-memory mapped logic in the module effectively putting it in a software-controlled power-saving state. The reset value of the MDIS bit is parameterized, with a default reset value of 0 .</p> <p>0 Enables the module clocks. 1 Allows external logic to disable the module clocks.</p>										
18 DIS_TXF	<p>Disable Transmit FIFO</p> <p>When the TX FIFO is disabled, the transmit part of the module operates as a simplified double-buffered SPI. This bit can be written only when the MDIS bit is cleared.</p> <p>0 TX FIFO is enabled. 1 TX FIFO is disabled.</p>										
19 DIS_RXF	Disable Receive FIFO										

*Table continues on the next page...*

**SPI\_MCR field descriptions (continued)**

Field	Description
	<p>When the RX FIFO is disabled, the receive part of the module operates as a simplified double-buffered SPI. This bit can only be written when the MDIS bit is cleared.</p> <p>0 RX FIFO is enabled. 1 RX FIFO is disabled.</p>
20 CLR_TXF	<p>Clear TX FIFO</p> <p>Flushes the TX FIFO. Writing a 1 to CLR_TXF clears the TX FIFO Counter. The CLR_TXF bit is always read as zero.</p> <p>0 Do not clear the TX FIFO counter. 1 Clear the TX FIFO counter.</p>
21 CLR_RXF	<p>Clear RX FIFO</p> <p>Flushes the RX FIFO. Writing a 1 to CLR_RXF clears the RX Counter. The CLR_RXF bit is always read as zero.</p> <p><b>NOTE:</b> After every RX FIFO clear operation (MCR [CLR_RXF] = 0b1) following a RX FIFO overflow (SR [RFOF] = 0b1) scenario, perform a single POP from the RX FIFO and discard the read data. The POP and discard operation must be completed before receiving a new incoming frame.</p> <p>0 Do not clear the RX FIFO counter. 1 Clear the RX FIFO counter.</p>
22–23 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
24–27 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
28 XSPI	<p>Extended SPI Mode</p> <p>This bit enables usage of CTARE (Command and Transfer Attribute Register Extended) Registers. CTARE registers allow the user to send up to 32 bit SPI frames. Command Cycling is also enabled which allows the user to send multiple Data Frames using a single Command Frame. When MCR[DIS_TXF] is asserted, the Extended SPI Mode cannot be used to transmit SPI frames which are more than 16 bits in size.</p> <p>0 Normal SPI Mode. Frame size can be up to 16 bits. Command Cycling is not available in this mode. 1 Extended SPI Mode. Up to 32 bit SPI Frames along with Command Cycling is Enabled.</p>
29 FCPCS	<p>Fast Continuous PCS Mode.</p> <p>This bit enables the masking of “After SCK (<math>t_{ASC}</math>)” and “PCS to SCK (<math>t_{CSC}</math>)” delays when operating in Continuous PCS mode. This masking is not available if Continuous SCK mode is enabled. The individual delay masks are selected via bits MASC and MCSC of the PUSH register. The firmware should select appropriate masks when providing continuous frames via the PUSH register.</p> <p>0 Normal or Slow Continuous PCS mode. Masking of delays is disabled. 1 Fast Continuous PCS mode. Delays masked via control bits in PUSH register.</p>
30 PES	<p>Parity Error Stop</p> <p>Controls SPI operation when a parity error is detected in a received SPI frame.</p> <p>0 SPI frame transmission continues. 1 SPI frame transmission stops.</p>

Table continues on the next page...

**SPI\_MCR field descriptions (continued)**

Field	Description
31 HALT	Halt  The HALT bit starts and stops frame transfers. See <a href="#">Start and Stop of Module transfers</a>  0 Start transfers. 1 Stop transfers.

**34.4.2 Transfer Count Register (SPI\_TCR)**

TCR contains a counter that indicates the number of SPI transfers made. The transfer counter is intended to assist in queue management. Do not write the TCR when the module is in the Running state.

Address: 210\_0000h base + 8h offset = 210\_0008h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SPI\_TCR field descriptions**

Field	Description
0–15 SPI_TCNT	SPI Transfer Counter  Counts the number of SPI transfers the module makes. The SPI_TCNT field increments every time the last bit of an SPI frame is transmitted. A value written to SPI_TCNT presets the counter to that value. SPI_TCNT is reset to zero at the beginning of the frame when the CTCNT field is set in the executing SPI command. The Transfer Counter wraps around; incrementing the counter past 65535 resets the counter to zero.
16–31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

**34.4.3 Clock and Transfer Attributes Register (In Master Mode) (SPI\_CTARn)**

CTAR registers are used to define different transfer attributes. Do not write to the CTAR registers while the module is in the Running state.

In Master mode, the CTAR registers define combinations of transfer attributes such as frame size, clock phase and polarity, data bit ordering, baud rate, and various delays.

## Memory Map/Register Definition

When the module is configured as a SPI master, the CTAS field in the command portion of the TX FIFO entry selects which of the CTAR registers is used.

Address: 210\_0000h base + Ch offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	DBR	FMSZ			CPOL	CPHA	LSBFE		PCSSCK	PASC		PDT	PBR			
W																
Reset	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CSSCK				ASC				DT			BR				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SPI\_CTARn field descriptions

Field	Description
0 DBR	<p>Double Baud Rate</p> <p>Doubles the effective baud rate of the Serial Communications Clock (SCK). This field is used only in master mode. It effectively halves the Baud Rate division ratio, supporting faster frequencies, and odd division ratios for the Serial Communications Clock (SCK). When the DBR bit is set, the duty cycle of the Serial Communications Clock (SCK) depends on the value in the Baud Rate Prescaler and the Clock Phase bit as listed in the following table. See the BR field description for details on how to compute the baud rate.</p>
1–4 FMSZ	<p>Frame Size</p> <p>The number of bits transferred per frame is equal to the FMSZ value plus 1. Regardless of the transmission mode, the minimum valid frame size value is 4.</p>

Table 34-6. SPI SCK Duty Cycle

DBR	CPHA	PBR	SCK Duty Cycle
0	any	any	50/50
1	0	00	50/50
1	0	01	33/66
1	0	10	40/60
1	0	11	43/57
1	1	00	50/50
1	1	01	66/33
1	1	10	60/40
1	1	11	57/43

0 The baud rate is computed normally with a 50/50 duty cycle.

1 The baud rate is doubled with the duty cycle depending on the Baud Rate Prescaler.

Table continues on the next page...

**SPI\_Ctar<sub>n</sub> field descriptions (continued)**

Field	Description
5 CPOL	<p>Clock Polarity</p> <p>Selects the inactive state of the Serial Communications Clock (SCK). For successful communication between serial devices, the devices must have identical clock polarities. When the Continuous Selection Format is selected, switching between clock polarities without stopping the module can cause errors in the transfer due to the peripheral device interpreting the switch of clock polarity as a valid clock edge.</p> <p><b>NOTE:</b> In case of Continuous SCK mode, when the module goes in low power mode(disabled), inactive state of SCK is not guaranteed.</p> <ul style="list-style-type: none"> <li>0 The inactive state value of SCK is low.</li> <li>1 The inactive state value of SCK is high.</li> </ul>
6 CPHA	<p>Clock Phase</p> <p>Selects which edge of SCK causes data to change and which edge causes data to be captured. This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the bit value is ignored and the transfers are done as if the CPHA bit is set to 1.</p> <ul style="list-style-type: none"> <li>0 Data is captured on the leading edge of SCK and changed on the following edge.</li> <li>1 Data is changed on the leading edge of SCK and captured on the following edge.</li> </ul>
7 LSBFE	<p>LSB First</p> <p>Specifies whether the LSB or MSB of the frame is transferred first.</p> <ul style="list-style-type: none"> <li>0 Data is transferred MSB first.</li> <li>1 Data is transferred LSB first.</li> </ul>
8–9 PCSSCK	<p>PCS to SCK Delay Prescaler</p> <p>Selects the prescaler value for the delay between assertion of PCS and the first edge of the SCK. See the CSSCK field description for information on how to compute the PCS to SCK Delay. Refer <a href="#">PCS to SCK Delay (<math>t_{csc}</math>)</a> for more details.</p> <ul style="list-style-type: none"> <li>00 PCS to SCK Prescaler value is 1.</li> <li>01 PCS to SCK Prescaler value is 3.</li> <li>10 PCS to SCK Prescaler value is 5.</li> <li>11 PCS to SCK Prescaler value is 7.</li> </ul>
10–11 PASC	<p>After SCK Delay Prescaler</p> <p>Selects the prescaler value for the delay between the last edge of SCK and the negation of PCS. See the ASC field description for information on how to compute the After SCK Delay. Refer <a href="#">After SCK Delay (<math>t_{asc}</math>)</a> for more details.</p> <ul style="list-style-type: none"> <li>00 Delay after Transfer Prescaler value is 1.</li> <li>01 Delay after Transfer Prescaler value is 3.</li> <li>10 Delay after Transfer Prescaler value is 5.</li> <li>11 Delay after Transfer Prescaler value is 7.</li> </ul>
12–13 PDT	<p>Delay after Transfer Prescaler</p> <p>Selects the prescaler value for the delay between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame. The PDT field is only used in master mode. See the DT field description for details on how to compute the Delay after Transfer. Refer <a href="#">Delay after Transfer (<math>t_{dt}</math>)</a> for more details.</p>

Table continues on the next page...

**SPI\_Ctar<sub>n</sub> field descriptions (continued)**

Field	Description																																		
	<p>00 Delay after Transfer Prescaler value is 1.      01 Delay after Transfer Prescaler value is 3.      10 Delay after Transfer Prescaler value is 5.      11 Delay after Transfer Prescaler value is 7.</p>																																		
14–15 PBR	<p>Baud Rate Prescaler</p> <p>Selects the prescaler value for the baud rate. This field is used only in master mode. The baud rate is the frequency of the SCK. The protocol clock is divided by the prescaler value before the baud rate selection takes place. See the BR field description for details on how to compute the baud rate.</p> <p>00 Baud Rate Prescaler value is 2.      01 Baud Rate Prescaler value is 3.      10 Baud Rate Prescaler value is 5.      11 Baud Rate Prescaler value is 7.</p>																																		
16–19 CSSCK	<p>PCS to SCK Delay Scaler</p> <p>Selects the scaler value for the PCS to SCK delay. This field is used only in master mode. The PCS to SCK Delay is the delay between the assertion of PCS and the first edge of the SCK. The delay is a multiple of the protocol clock period, and it is computed according to the following equation:</p> $t_{csc} = (1/f_p) \times PCSSCK \times CSSCK$ <p>The following table lists the delay scaler values.</p>																																		
	<b>Table 34-7. Delay Scaler Encoding</b>																																		
	<table border="1"> <thead> <tr> <th>Field Value</th><th>Delay Scaler Value</th></tr> </thead> <tbody> <tr><td>0000</td><td>2</td></tr> <tr><td>0001</td><td>4</td></tr> <tr><td>0010</td><td>8</td></tr> <tr><td>0011</td><td>16</td></tr> <tr><td>0100</td><td>32</td></tr> <tr><td>0101</td><td>64</td></tr> <tr><td>0110</td><td>128</td></tr> <tr><td>0111</td><td>256</td></tr> <tr><td>1000</td><td>512</td></tr> <tr><td>1001</td><td>1024</td></tr> <tr><td>1010</td><td>2048</td></tr> <tr><td>1011</td><td>4096</td></tr> <tr><td>1100</td><td>8192</td></tr> <tr><td>1101</td><td>16384</td></tr> <tr><td>1110</td><td>32768</td></tr> <tr><td>1111</td><td>65536</td></tr> </tbody> </table>	Field Value	Delay Scaler Value	0000	2	0001	4	0010	8	0011	16	0100	32	0101	64	0110	128	0111	256	1000	512	1001	1024	1010	2048	1011	4096	1100	8192	1101	16384	1110	32768	1111	65536
Field Value	Delay Scaler Value																																		
0000	2																																		
0001	4																																		
0010	8																																		
0011	16																																		
0100	32																																		
0101	64																																		
0110	128																																		
0111	256																																		
1000	512																																		
1001	1024																																		
1010	2048																																		
1011	4096																																		
1100	8192																																		
1101	16384																																		
1110	32768																																		
1111	65536																																		
	Refer <a href="#">PCS to SCK Delay (<math>t_{csc}</math>)</a> for more details.																																		
20–23 ASC	After SCK Delay Scaler																																		

*Table continues on the next page...*

**SPI\_Ctar<sub>n</sub> field descriptions (continued)**

Field	Description
	<p>Selects the scaler value for the After SCK Delay. This field is used only in master mode. The After SCK Delay is the delay between the last edge of SCK and the negation of PCS. The delay is a multiple of the protocol clock period, and it is computed according to the following equation:</p> $t_{ASC} = (1/f_P) \times PASC \times ASC$ <p>See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values. Refer <a href="#">After SCK Delay (<math>t_{ASC}</math>)</a> for more details.</p>
24–27 DT	<p>Delay After Transfer Scaler</p> <p>Selects the Delay after Transfer Scaler. This field is used only in master mode. The Delay after Transfer is the time between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame.</p> <p>In the Continuous Serial Communications Clock operation, the DT value is fixed to one SCK clock period. The Delay after Transfer is a multiple of the protocol clock period, and it is computed according to the following equation:</p> $t_{DT} = (1/f_P) \times PDT \times DT$ <p>See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values.</p>
28–31 BR	<p>Baud Rate Scaler</p> <p>Selects the scaler value for the baud rate. This field is used only in master mode. The prescaled protocol clock is divided by the Baud Rate Scaler to generate the frequency of the SCK. The baud rate is computed according to the following equation:</p> $\text{SCK baud rate} = (f_P / PBR) \times [(1+DBR)/BR]$ <p>The following table lists the baud rate scaler values.</p>

**Table 34-8. Baud Rate Scaler**

CTARn[BR]	Baud Rate Scaler Value
0000	2
0001	4
0010	6
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	512
1010	1024
1011	2048
1100	4096
1101	8192
1110	16384
1111	32768

### 34.4.4 Status Register (SPI\_SR)

SR contains status and flag bits. The bits reflect the status of the module and indicate the occurrence of events that can generate interrupt or DMA requests. Software can clear flag bits in the SR by writing a 1 to them. Writing a 0 to a flag bit has no effect. This register may not be writable in Module Disable mode due to the use of power saving mechanisms.

Address: 210\_0000h base + 2Ch offset = 210\_002Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TCF	TXRXS	0	EOQF	0	0	TFFF	BSYF	CMDTCF	0	SPEF	0	RFOF	TFIWF	RFDF	CMDFFFF
W	w1c			w1c			w1c		w1c		w1c		w1c		w1c	
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TXCTR				TXNXTPTR				RXCTR				POPNXTPTR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SPI\_SR field descriptions

Field	Description
0 TCF	<p>Transfer Complete Flag</p> <p>Indicates that all bits in a frame have been shifted out. TCF remains set until it is cleared by writing a 1 to it.</p> <p>0 Transfer not complete. 1 Transfer complete.</p>

Table continues on the next page...

**SPI\_SR field descriptions (continued)**

Field	Description
1 TXRXS	<p>TX and RX Status</p> <p>Reflects the run status of the module.</p> <p>0 Transmit and receive operations are disabled (The module is in Stopped state). 1 Transmit and receive operations are enabled (The module is in Running state).</p>
2 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
3 EOQF	<p>End of Queue Flag</p> <p>Indicates that the last entry in a queue has been transmitted when the module is in Master mode. The EOQF bit is set when the TX FIFO entry has the EOQ bit set in the command halfword and the end of the transfer is reached. The EOQF bit remains set until cleared by writing a 1 to it. When the EOQF bit is set, the TXRXS bit is automatically cleared.</p> <p>0 EOQ is not set in the executing command. 1 EOQ is set in the executing SPI command.</p>
4 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
5 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
6 TFFF	<p>Transmit FIFO Fill Flag</p> <p>Indicates whether there is an available location to be filled in the FIFO. Either a DMA request or an interrupt indication can be used to add another entry to the FIFO. Note that this bit is set if at least one location is free in the FIFO. The TFFF bit can be cleared by writing 1 to it or by acknowledgement from the DMA controller to the TX FIFO full request, when the TX FIFO is full.</p> <p>If FIFO is filled manually, and not by DMA see <a href="#">Transmit FIFO Fill Interrupt or DMA Request</a></p> <p><b>NOTE:</b> The reset value of this bit is 0 when the module is disabled,(MCR[MDIS]=1).</p> <p>0 TX FIFO is full. 1 TX FIFO is not full.</p>
7 BSYF	<p>Busy Flag.</p> <p>This bit is valid only when SPI_MCR[XSPI] is enabled. Indicates that the current Command Frame is being used for transmitting multiple data frames. This bit is not set for the last Data Frame of a Cyclic command Transfer or when SPI_CTARE[DTCP] = 1. Refer <a href="#">Command First In First Out (CMD FIFO) Buffering Mechanism</a> for more details.</p> <p>0 No Cyclic Command Transfer in Progress. 1 Cyclic Command Transfer is in progress. Current Data Frame is not the last data frame for on-going cyclic command transfer..</p>
8 CMDTCF	<p>Command Transfer Complete Flag.</p> <p>Indicates that the last Data frame for the current Cyclic Command has been transmitted. Hence this bit is set only for the last Data Frame of a Cyclic Command Transfer or when SPI_CTARE[DTCP] = 1. The bit remains set until it is cleared by writing a '1' to it.</p> <p>0 Data Transfer by current Command not complete. 1 Data Transfer by current Command is complete.</p>

*Table continues on the next page...*

**SPI\_SR field descriptions (continued)**

Field	Description
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 SPEF	SPI Parity Error Flag  Indicates that a SPI frame with parity error had been received. The bit remains set until it is cleared by writing a 1 to it.  0 No parity error. 1 Parity error has occurred.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 RFOF	Receive FIFO Overflow Flag  Indicates an overflow condition in the RX FIFO. The field is set when the RX FIFO and shift register are full and a transfer is initiated. The bit remains set until it is cleared by writing a 1 to it.  0 No Rx FIFO overflow. 1 Rx FIFO overflow has occurred.
13 TFIWF	Transmit FIFO Invalid Write Flag  Indicates Data Write on TX FIFO while CMD FIFO is empty. Without a Command, the Data entries present in TXFIFO are invalid. This bit remains set until it is cleared by writing a '1' to it.  0 No Invalid Data present in TX FIFO. 1 Invalid Data present in TX FIFO since CMD FIFO is empty.
14 RFDF	Receive FIFO Drain Flag  Provides a method for the module to request that entries be removed from the RX FIFO. Note that this bit is set if at least one location can be read from the FIFO. The RFDF bit can be cleared by acknowledgement from the DMA controller when the RX FIFO is empty.  0 RX FIFO is empty. 1 This bit auto-clears on every RXFR read performed.
15 CMDFFF	Command FIFO Fill Flag  Indicates whether there is an available location to be filled in the FIFO. Either a DMA request or an interrupt indication can be used to add another entry to the FIFO. Note that this bit is set if at least one location is free in the FIFO. The CMDFFF is cleared by writing a '1' to it or by acknowledgement from the DMA controller to the CMD FIFO full request.  0 CMD FIFO is full. 1 CMD FIFO is not full.
16–19 TXCTR	TX FIFO Counter  Indicates the number of valid entries in the TX FIFO. The TXCTR is incremented every time the PUSHR is written. The TXCTR is decremented every time an SPI command is executed and the SPI data is transferred to the shift register.
20–23 TXNXTPTR	Transmit Next Pointer  Indicates which TX FIFO entry is transmitted during the next transfer. The TXNXTPTR field is updated every time SPI data is transferred from the TX FIFO to the shift register.

*Table continues on the next page...*

**SPI\_SR field descriptions (continued)**

Field	Description
	<b>NOTE:</b> When TX FIFO is cleared by setting SPI_MCR[CLR_TXF] to 1, this field does not update immediately to 0. Only when the next transfer starts, this field reflects the latest value TXNXTPTR =1.
24–27 RXCTR	RX FIFO Counter  Indicates the number of entries in the RX FIFO. The RXCTR is decremented every time the POPR is read. The RXCTR is incremented every time data is transferred from the shift register to the RX FIFO.
28–31 POPNXTPTR	Pop Next Pointer  Contains a pointer to the RX FIFO entry to be returned when the POPR is read. The POPNXTPTR is updated when the POPR is read.

**34.4.5 DMA/Interrupt Request Select and Enable Register (SPI\_RSER)**

RSER controls DMA and interrupt requests. Do not write to the RSER while the module is in the Running state.

Address: 210\_0000h base + 30h offset = 210\_0030h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	TCF_RE	CMDFFF_RE	Reserved	EOQF_RE	Reserved	Reserved	Reserved	TFFF_RE	CMDTCE_RE	Reserved	SPEF_RE	Reserved	RFOF_RE	TFIWF_RE	RFDF_RE	RFDF_DIRS
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	CMDFFF_DIRS	Reserved	0													
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPI\_RSER field descriptions**

Field	Description
0 TCF_RE	Transmission Complete Request Enable  Enables TCF flag in the SR to generate an interrupt request.  0 TCF interrupt requests are disabled. 1 TCF interrupt requests are enabled.

Table continues on the next page...

**SPI\_RSER field descriptions (continued)**

Field	Description
1 CMDFFF_RE	<p>Command FIFO Fill Flag Request Enable.</p> <p>Enables the CMDFFF flag in the SR to generate a request. The CMDFFF_DIRS bit selects between generating an interrupt request or a DMA request.</p> <p>0 CMDFFF interrupts or DMA requests are disabled. 1 CMDFFF interrupts or DMA requests are enabled.</p>
2 Reserved	<p>Always write the reset value to this field.</p> <p>This field is reserved.</p>
3 EOQF_RE	<p>Finished Request Enable</p> <p>Enables the EOQF flag in the SR to generate an interrupt request.</p> <p>0 EOQF interrupt requests are disabled. 1 EOQF interrupt requests are enabled.</p>
4 Reserved	<p>This field is reserved.</p>
5 Reserved	<p>Always write the reset value to this field.</p> <p>This field is reserved.</p>
6 TFFF_RE	<p>Transmit FIFO Fill Request Enable</p> <p>Enables the TFFF flag in the SR to generate a request. The TFFF_DIRS bit selects between generating an interrupt request or a DMA request.</p> <p>0 TFFF interrupts or DMA requests are disabled. 1 TFFF interrupts or DMA requests are enabled.</p>
7 TFFF_DIRS	<p>Transmit FIFO Fill DMA or Interrupt Request Select</p> <p>Selects between generating a DMA request or an interrupt request. When SR[TFFF] and RSER[TFFF_RE] are set, this field selects between generating an interrupt request or a DMA request.</p> <p>0 TFFF flag generates interrupt requests. 1 TFFF flag generates DMA requests.</p>
8 CMDTCF_RE	<p>Command Transmission Complete Request Enable.</p> <p>The CMDTCF_RE bit enables CMDTCF flag in the SR to generate an interrupt request.</p> <p>0 CMDTCF interrupt requests are disabled. 1 CMDTCF interrupt requests are enabled.</p>
9 Reserved	<p>Always write the reset value to this field.</p> <p>This field is reserved.</p>
10 SPEF_RE	<p>SPI Parity Error Request Enable</p> <p>Enables the SPEF flag in the SR to generate an interrupt request.</p> <p>0 SPEF interrupt requests are disabled. 1 SPEF interrupt requests are enabled.</p>

*Table continues on the next page...*

**SPI\_RSER field descriptions (continued)**

Field	Description
11 Reserved	Always write the reset value to this field.  This field is reserved.
12 RFOF_RE	Receive FIFO Overflow Request Enable  Enables the RFOF flag in the SR to generate an interrupt request.  0 RFOF interrupt requests are disabled. 1 RFOF interrupt requests are enabled.
13 TFIWF_RE	Transmit FIFO Invalid Write Request Enable.  Enables the TFIWF flag in the SR to generate an interrupt request.  0 TFIWF interrupt requests are disabled. 1 TFIWF interrupt requests are enabled.
14 RFDF_RE	Receive FIFO Drain Request Enable  Enables the RFDF flag in the SR to generate a request. The RFDF_DIRS bit selects between generating an interrupt request or a DMA request.  0 RFDF interrupt or DMA requests are disabled. 1 RFDF interrupt or DMA requests are enabled.
15 RFDF_DIRS	Receive FIFO Drain DMA or Interrupt Request Select  Selects between generating a DMA request or an interrupt request. When the RFDF flag bit in the SR is set, and the RFDF_RE bit in the RSER is set, the RFDF_DIRS bit selects between generating an interrupt request or a DMA request.  0 Interrupt request. 1 DMA request.
16 CMDFFF_DIRS	Command FIFO Fill DMA or Interrupt Request Select  Selects between generating a DMA request or an interrupt request. When the CMDFFF flag bit in the SR is set, and the CMDFFF_RE bit in the RSER is set, the CMDFFF_DIRS bit selects between generating an interrupt indication or a DMA request.  0 CMDFFF flag generates interrupt requests. 1 CMDFFF flag generates DMA requests.
17 Reserved	Always write the reset value to this field.  This field is reserved.
18–31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

### 34.4.6 PUSH TX FIFO Register In Master Mode (SPI\_PUSHR)

Specifies data to be transferred to the TX FIFO and CMD FIFO. User must write 16-bits data into TXDATA field. An 8- or 16-bit write access to the TXDATA field transfers 16 bits of data bus to the TX FIFO. A write access to the command fields transfers the 16 bits of command information to the CMD FIFO. In Master mode, the register transfers 16 bits of data to the TX FIFO and 16 bits of command information to the CMD FIFO.

If Extended SPI Mode is disabled (MCR[XSPI]), the TX FIFO and CMD FIFO must be filled simultaneously. In other words, you must perform write accesses to both the data and command fields for every PUSHR operation. With Extended SPI Mode disabled and both the TX FIFO and CMD FIFO are written to and read from simultaneously, they behave as a single 32 bit FIFO. When Extended SPI mode is enabled (MCR[XSPI]), the TX FIFO and CMD FIFO can be written to independently.

A read access of PUSHR returns the topmost TX FIFO and CMD FIFO entries concatenated.

When the module is disabled, writing to this register does not update the FIFO. Therefore, any reads performed while the module is disabled return the last PUSHR write performed while the module was still enabled.

Address: 210\_0000h base + 34h offset = 210\_0034h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CONT		CTAS		EOQ	CTCNT	PE_MASC	PP_MCSC								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									TXDATA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SPI\_PUSHR field descriptions

Field	Description
0 CONT	Continuous Peripheral Chip Select Enable  Selects a continuous selection format. The bit is used in SPI Master mode. The bit enables the selected PCS signals to remain asserted between transfers.

*Table continues on the next page...*

**SPI\_PUSHR field descriptions (continued)**

Field	Description																
	<p>0 Return PCSn signals to their inactive state between transfers. 1 Keep PCSn signals asserted between transfers.</p>																
1–3 CTAS	<p>Clock and Transfer Attributes Select</p> <p>Selects which CTAR to use in master mode to specify the transfer attributes for the associated SPI frame. See the chip specific section for details to determine how many CTARs this device has. You should not program a value in this field for a register that is not present.</p> <table> <tr><td>000</td><td>CTAR0</td></tr> <tr><td>001</td><td>CTAR1</td></tr> <tr><td>010</td><td>CTAR2</td></tr> <tr><td>011</td><td>CTAR3</td></tr> <tr><td>100</td><td>Reserved</td></tr> <tr><td>101</td><td>Reserved</td></tr> <tr><td>110</td><td>Reserved</td></tr> <tr><td>111</td><td>Reserved</td></tr> </table>	000	CTAR0	001	CTAR1	010	CTAR2	011	CTAR3	100	Reserved	101	Reserved	110	Reserved	111	Reserved
000	CTAR0																
001	CTAR1																
010	CTAR2																
011	CTAR3																
100	Reserved																
101	Reserved																
110	Reserved																
111	Reserved																
4 EOQ	<p>End Of Queue</p> <p>Host software uses this bit to signal to the module that the current SPI transfer is the last in a queue. At the end of the transfer, the EOQF bit in the SR is set.</p> <p>0 The SPI data is not the last data to transfer. 1 The SPI data is the last data to transfer.</p>																
5 CTCNT	<p>Clear Transfer Counter</p> <p>Clears the TCNT field in the TCR register. The TCNT field is cleared before the module starts transmitting the current SPI frame.</p> <p>0 Do not clear the TCR[TCNT] field. 1 Clear the TCR[TCNT] field.</p>																
6 PE_MASC	<p>Parity Enable or Mask T<sub>ASC</sub> delay in the current frame</p> <p>PE – This bit enables parity bit transmission and parity reception check for the SPI frame. MASC - The current frame has the “after SCK” delay masked if this bit is asserted. See <a href="#">Fast Continuous Selection Format</a> for more details.</p> <p><b>NOTE:</b> This bit is used as Mask T<sub>ASC</sub> in the Fast Continuous PCS mode when MCR[FCPCS] is set.</p> <p>0 PE - No parity bit included/checked. MASC - T<sub>ASC</sub> delay is not masked and the current frame has the after SCK delay. 1 PE - Parity bit is transmitted instead of the last data bit in the frame; parity is checked for the received frame. MASC - T<sub>ASC</sub> delay is masked in the current frame.</p>																
7 PP_MCSC	<p>Parity Polarity or Mask T<sub>csc</sub> delay in the next frame</p> <p>PP - It controls the polarity of the parity bit transmitted and checked. MCSC - The next frame has the “PCS to SCK” delay masked if this bit is asserted. See <a href="#">Fast Continuous Selection Format</a> for more details.</p> <p><b>NOTE:</b> This bit is used as Mask T<sub>csc</sub> in the Fast Continuous PCS mode when MCR[FCPCS] is set.</p>																

*Table continues on the next page...*

**SPI\_PUSHR field descriptions (continued)**

Field	Description										
	<p>0 PP - Even Parity: the number of 1 bits in the transmitted frame is even. The SR[SPEF] bit is set if the number of 1 bits is odd in the received frame.</p> <p>MCSC - T<sub>CSC</sub> delay is not masked and the next frame has the PCS to SCK delay.</p> <p>1 PP - Odd Parity: the number of 1 bits in the transmitted frame is odd. The SR[SPEF] bit is set if the number of 1 bits is even in the received frame.</p> <p>MCSC - T<sub>CSC</sub> delay is masked in the next frame.</p>										
8–11 Reserved	Always write the reset value to this field.  This field is reserved.										
12–15 PCS	Select which PCS signals are to be asserted for the transfer. Refer to the chip-specific SPI information for the number of PCS signals used in this chip.										
	<b>Table 34-9. PCS signal and bit position mapping</b>										
	<table border="1"> <thead> <tr> <th>Signal</th><th>Bit position</th></tr> </thead> <tbody> <tr> <td>PCS0</td><td>15</td></tr> <tr> <td>PCS1</td><td>14</td></tr> <tr> <td>PCS2</td><td>13</td></tr> <tr> <td>PCS3</td><td>12</td></tr> </tbody> </table>	Signal	Bit position	PCS0	15	PCS1	14	PCS2	13	PCS3	12
Signal	Bit position										
PCS0	15										
PCS1	14										
PCS2	13										
PCS3	12										
	<p>0 Negate the PCS[x] signal.</p> <p>1 Assert the PCS[x] signal.</p>										
16–31 TXDATA	<p>Transmit Data</p> <p>Holds SPI data to be transferred according to the associated SPI command.</p>										

**34.4.7 POP RX FIFO Register (SPI\_POPR)**

POPR is used to read the RX FIFO. Eight- or sixteen-bit read accesses to the POPR have the same effect on the RX FIFO as 32-bit read accesses. A write to this register will generate a Transfer Error.

Address: 210\_0000h base + 38h offset = 210\_0038h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SPI\_POPR field descriptions**

Field	Description
0–31 RXDATA	Received Data  Contains the SPI data from the RX FIFO entry to which the Pop Next Data Pointer points.

**34.4.8 Transmit FIFO Registers (SPI\_TXFRn)**

TXFRn registers provide visibility into the TX FIFO for debugging purposes. Each register is an entry in the TX FIFO. The registers are read-only and cannot be modified. Reading the TXFRx registers does not alter the state of the TX FIFO. When the module is operating in Extended SPI mode, reading TXFRn registers is invalid.

Address: 210\_0000h base + 3Ch offset + (4d × i), where i=0d to 15d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
	TXCMD_TCDATA																TXDATA															
W																																

**SPI\_TXFRn field descriptions**

Field	Description
0–15 TXCMD_TCDATA	Transmit Command or Transmit Data  In Master mode the TXCMD field contains the command that sets the transfer attributes for the SPI data.
16–31 TXDATA	Transmit Data  Contains the SPI data to be shifted out.

### 34.4.9 Receive FIFO Registers (SPI\_RXFR $n$ )

RXFR $n$  provide visibility into the RX FIFO for debugging purposes. Each register is an entry in the RX FIFO. The RXFR registers are read-only. Reading the RXFRx registers does not alter the state of the RX FIFO. The field MCR[MDIS] must be 0 when RXFR is read.

Address: 210\_0000h base + 7Ch offset + (4d × i), where i=0d to 15d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### SPI\_RXFR $n$ field descriptions

Field	Description
0–31 RXDATA	Receive Data  Contains the received SPI data.

### 34.4.10 Clock and Transfer Attributes Register Extended (SPI\_CTARE $n$ )

CTARE registers are used to define the extended transfer attributes for an SPI frame. These registers are valid only when Extended SPI mode is enabled (MCR[XSPI]).

When the module is configured as a SPI master, the CTAS field in CMD FIFO entry selects which of the CTARE registers is used.

Address: 210\_0000h base + 11Ch offset + (4d × i), where i=0d to 3d

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R					0											
W																DTCP
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**SPI\_CTAREn field descriptions**

Field	Description
0–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15 FMSZE	Frame Size Extended  This field is valid only when MCR[XSPI] is set. This field concatenated with CTAR[FMSZ] defines the Frame size of the SPI frames to be transmitted. Effective frame size would be the concatenation of {CTARE[FMSZE], CTAR[FMSZ]} plus 1.  0 Default Mode. Up to 16 bit SPI frames can be transferred. 1 Up to 32 bit SPI frames can be transferred. Each Frame transfer will be a result of 2 simultaneous TX FIFO Pop Operation.
16–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21–31 DTCP	Data Transfer Count Preload  This field is valid only when SPIx_MCR[XSPI] is set. This field defines the number of data frames (whose size is defined by CTARE[FMSZE] and CTAR[FMSZ]) to be transmitted using the Command frame which selected this SPIx_CTARE register. The value 0 is reserved and should not be written in this field. The default value of this field is 1.

### 34.4.11 Status Register Extended (SPI\_SREX)

The register contains status fields. The fields reflect the status of the module and indicate the occurrence of events. This register is not writable.

Address: 210\_0000h base + 13Ch offset = 210\_013Ch

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	TXCTR <sup>4</sup>		0	RXCTR <sup>4</sup>		0		CMDCTR			CMDNXTPTR				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SPI\_SREX field descriptions

Field	Description
0–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 TXCTR4	TX FIFO Counter[4]  This bit is an extension of SR[TXCTR]. The concatenated field {TXCTR4, TXCTR} indicates the number of valid entries in the TX FIFO. This field is incremented every time the PUSHR is written. And this field is decremented every time an SPI command is executed and the SPI data is transferred to the shift register.
18–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

### SPI\_SREX field descriptions (continued)

Field	Description
20 RXCTR4	RX FIFO Counter[4]  This bit is an extension of SR[RXCTR]. The concatenated field {RXCTR4, RXCTR} indicates the number of entries in the RX FIFO. This field is decremented every time the POPR is read. And this field is incremented every time data is transferred from the shift register to the RX FIFO.
21–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–27 CMDCTR	CMD FIFO Counter  Indicates the number of entries in the CMD FIFO. The CMDCTR is incremented every time the command part of PUSHR is written. The CMDCTR is decremented every time a SPI command is executed (all data frames due to current command frame have been transmitted).
28–31 CMDNXTPTR	Command Next Pointer  Indicates which CMD FIFO Entry is used during the next transfer. The CMDNXTPTR field is updated every time SPI data due to current command have been transmitted.

## 34.5 Functional description

The module supports full-duplex, synchronous serial communications between chips and peripheral devices. The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes.

The module has the following configuration

- The SPI Configuration in which the module operates as a basic SPI or a queued SPI.

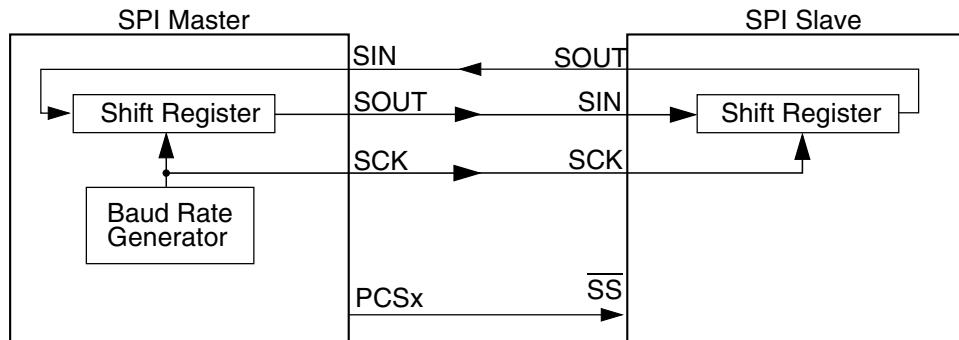
The DCONF field in the Module Configuration Register (MCR) determines the module Configuration. SPI configuration is selected when DCONF within SPIx\_MCR is 2b00.

The CTARn registers hold clock and transfer attributes. The SPI configuration allows to select which CTAR to use on a frame by frame basis by setting a field in the SPI command. The Extended SPI Mode (SPIx\_MCR[XSPI]) further allows the usage CTAREn (CTARn Extended) registers which allows the user to send multiple data frames using a single command frame.

See [Clock and Transfer Attributes Register \(In Master Mode\) \(SPI\\_CTARn\)](#) for information on the fields of CTAR registers. See [Clock and Transfer Attributes Register Extended \(SPI\\_CTAREn\)](#) for information on the fields of CTARE registers.

Typical master to slave connections are shown in the following figure. When a data transfer operation is performed, data is serially shifted a predetermined number of bit positions. Because the modules are linked, data is exchanged between the master and the

slave. The data that was in the master shift register is now in the shift register of the slave, and vice versa. At the end of a transfer, the Transfer Control Flag(TCF) bit in the Shift Register(SR) is set to indicate a completed frame transfer.



**Figure 34-3. Serial protocol overview**

Generally, more than one slave device can be connected to the module master. 4 Peripheral Chip Select (PCS) signals of the module masters can be used to select which of the slaves to communicate with. Refer to the chip specific section for details on the number of PCS signals used in this chip.

The SPI configuration shares transfer protocol and timing properties which are described independently of the configuration in [Transfer formats](#). The transfer rate and delay settings are described in [Module baud rate and clock delay generation](#).

### 34.5.1 Start and Stop of module transfers

The module has two operating states: Stopped and Running. Both the states are independent of it's configuration. The default state of the module is Stopped. In the Stopped state, no serial transfers are initiated in Master mode. The Stopped state is also a safe state for writing the various configuration registers of the module without causing undetermined results. In the Running state serial transfers take place.

The TXRXS bit in the SR indicates the state of module. The bit is set if the module is in Running state.

The module starts or transitions to Running when all of the following conditions are true:

- SR[EOQF] bit is clear
- Chip is not in the Debug mode or the MCR[FRZ] bit is clear
- MCR[HALT] bit is clear

The module stops or transitions from Running to Stopped after the current frame when any one of the following conditions exist:

- SR[EOQF] bit is set
- Chip in the Debug mode and the MCR[FRZ] bit is set
- MCR[HALT] bit is set

State transitions from Running to Stopped occur on the next frame boundary if a transfer is in progress, or immediately if no transfers are in progress.

### 34.5.2 Serial Peripheral Interface SPI configuration

The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes. The module is in SPI configuration when the DCONF field in the MCR is 2b00. The SPI frames can be 32 bits long. The host CPU or a DMA controller transfers the SPI data from the external to the module RAM queues to a TX FIFO buffer. The received data is stored in entries in the RX FIFO buffer. The host CPU or the DMA controller transfers the received data from the RX FIFO to memory external to the module. The operation of FIFO buffers is described in the following sections:

- [Transmit First In First Out \(TX FIFO\) buffering mechanism](#)
- [Command First In First Out \(CMD FIFO\) Buffering Mechanism](#)
- [Receive First In First Out \(RX FIFO\) buffering mechanism](#)

The interrupt and DMA request conditions are described in [Interrupts/DMA requests](#).

In Master mode the module initiates and controls the transfer according to the fields of the executing SPI Command.

#### 34.5.2.1 Master mode

In SPI Master mode, the module initiates the serial transfers by controlling the SCK and the PCS signals. The executing SPI Command determines which CTARs will be used to set the transfer attributes and which PCS signals to assert. The command field also contains various bits that help with queue management and transfer protocol. See [PUSH TX FIFO Register In Master Mode \(SPI\\_PUSHR\)](#) for details on the SPI command fields. The data in the executing TX FIFO entry is loaded into the shift register and shifted out on the Serial Out (SOUT) pin. In SPI Master mode, each SPI frame to be transmitted has a command associated with it, allowing for transfer attribute control on a frame by frame basis. In Extended SPI Master Mode, multiple SPI frames can have a single command

associated with them allowing for efficient SPI frame transfers requiring common transfer attributes. In addition, the Extended SPI Mode allows for larger frame sizes of up to 32 bits.

### **34.5.2.2 FIFO disable operation**

The FIFO disable mechanisms allow SPI transfers without using the TX FIFO, CMD FIFO or RX FIFO. The module operates as a double-buffered simplified SPI when the FIFOs are disabled. The Transmit and Receive side of the FIFOs are disabled separately. Setting the MCR[DIS\_TXF] bit disables the TX FIFO and CMD FIFO, and setting the MCR[DIS\_RXF] bit disables the RX FIFO.

The FIFO disable mechanisms are transparent to the user and to host software. Transmit data and commands are written to the PUSH Register and received data is read from the POPR.

When the TX FIFO and CMD FIFO are disabled:

- SR[TFFF], SR[TFUF], SR[CMDFFF], SREX[CMDCTR] and SR[TXCTR] behave as if there is a one-entry FIFO
- The contents of TXFRs, SR[TXNXTPT] and SREX[CMDNXTPT] are undefined

Similarly, when the RX FIFO is disabled, the RFDF, RFOF, and RXCTR fields in the SR behave as if there is a one-entry FIFO, but the contents of the RXFR registers and POPNXTPT are undefined.

### **34.5.2.3 Transmit First In First Out (TX FIFO) buffering mechanism**

The TX FIFO functions as a buffer of SPI data for transmission. The TX FIFO holds 16 words, each consisting of SPI data. The number of entries in the TX FIFO is device-specific. SPI data is added to the TX FIFO by writing to the Data Field of module PUSH FIFO Register (PUSHR). TX FIFO entries can only be removed from the TX FIFO by being shifted out or by flushing the TX FIFO.

The TX FIFO Counter field (TXCTR) in the module Status Register (SR) indicates the number of valid entries in the TX FIFO. The TXCTR is updated every time a 8- or 16-bit write takes place to PUSHR[TXDATA] or SPI data is transferred into the shift register from the TX FIFO.

The TXNXTPT field indicates the TX FIFO Entry that will be transmitted during the next transfer. The TXFRn Registers are invalid in the Extended SPI Mode, since the TX FIFO and CMD FIFO can be used independently. The TXNXTPT field is incremented

every time SPI data is transferred from the TX FIFO to the shift register. The maximum value of the field is equal to the maximum implemented TXFR number and it rolls over after reaching the maximum.

### 34.5.2.3.1 Filling the TX FIFO

Host software or other intelligent blocks can add (push) entries to the TX FIFO and CMD FIFO by writing to the PUSHR. When the TX FIFO is not full, the TX FIFO Fill Flag (TFFF) in the SR is set. The TFFF bit is cleared when TX FIFO is full and the DMA controller indicates that a write to PUSHR is complete. Writing a '1' to the TFFF bit also clears it. The TFFF can generate a DMA request or an interrupt request. See [Transmit FIFO Fill Interrupt or DMA Request](#) for details.

The module ignores attempts to push data to a full TX FIFO, and the state of the TX FIFO does not change and no error condition is indicated.

### 34.5.2.3.2 Draining the TX FIFO

The TX FIFO entries are removed (drained) by shifting SPI data out through the shift register. Entries are transferred from the TX FIFO to the shift register and shifted out as long as there are valid entries in the TX FIFO. Every time an entry is transferred from the TX FIFO to the shift register, the TX FIFO Counter decrements by one. When Extended SPI Mode (SPIx\_MCR[XSPI]) is enabled, if the frame size of SPI Data to be transmitted is more than 16 bits, then it causes two Data entries to be popped from TX FIFO simultaneously which are transferred to the shift register. The first of the two popped entries forms the 16 least significant bits of the SPI frame to be transmitted. Such an operation also causes TX FIFO Counter to decrement by two. At the end of a transfer, the TCF bit in the SR is set to indicate the completion of a transfer. The TX FIFO is flushed by writing a '1' to the CLR\_TXF bit in MCR.

If an external bus master initiates a transfer with a module slave while the slave's TX FIFO is empty, the Transmit FIFO Underflow Flag (TFUF) in the slave's SR is set.

### 34.5.2.4 Command First In First Out (CMD FIFO) Buffering Mechanism

The CMD FIFO functions as a buffer of SPI command used for SPI data transmission. When Extended SPI Mode (MCR[XSPI]) is disabled, the TX FIFO and CMD FIFO must be filled together, i.e. write enables should be given for both the Data and Command fields while performing a PUSHR operation. When Extended SPI Mode (MCR[XSPI]) is enabled, the TX FIFO and CMD FIFO can be filled independently.

The CMD FIFO holds 16 words, each representing SPI command fields. The number of entries in the CMD FIFO is device-specific. SPI Command is added to the CMD FIFO by writing to the command field of SPI PUSH FIFO Register (PUSHR). CMD FIFO entries can only be removed from the CMD FIFO by being shifted out (to help transmit SPI data) or by flushing the CMD FIFO.

When Extended SPI Mode (MCR[XSPI]) is disabled, every CMD FIFO entry has a corresponding single TX FIFO entry attached to it because both these FIFO's are filled simultaneously.

When Extended SPI Mode (MCR[XSPI]) is enabled, every CMD FIFO entry can have multiple TX FIFO entries attached to it. Thus a single CMD FIFO entry can be used to transmit multiple TX FIFO entries. The CTARE[DTCP] field decides the number of SPI Data Frames having frame size as {FMSZE, FMSZ} to be transmitted using the current Command Entry. The CTAR/CTARE registers pointed by the CTAS field in the Command frame gives the FMSZ and FMSZE fields respectively. The time for which a command entry is in use is known as a Command Cycle. The Busy Flag SR[BSYF] is asserted for the duration of the Command Cycle except for the last SPI frame in the Command Cycle.

The CMD FIFO Counter field (CMDCTR) in the SPIStatus Register (SR) indicates the number of valid entries in the CMD FIFO. The CMDCTR field is updated every time a 8- or 16-bit write takes place on the lower half of SPI\_PUSHR or SPI data is transferred into the shift register from the TX FIFO.

The TXFRn Registers are invalid in the Extended SPI Mode, since the TX FIFO and CMD FIFO can be used independently. The CMDNXTPTR field indicates which CMD FIFO Entry will be used during the next command cycle. The CMDNXTPTR field is incremented every time the last SPI data in the command cycle is transferred from the TX FIFO to the shift register and it rolls over after reaching the maximum.

### **34.5.2.5 Receive First In First Out (RX FIFO) buffering mechanism**

The RX FIFO functions as a buffer for data received on the SIN pin. The RX FIFO holds 16 received SPI data frames. The number of entries in the RX FIFO is device-specific. SPI data is added to the RX FIFO at the completion of a transfer when the received data in the shift register is transferred into the RX FIFO. SPI data are removed (popped) from the RX FIFO by reading the module POP RX FIFO Register (POPR). RX FIFO entries can only be removed from the RX FIFO by reading the POPR or by flushing the RX FIFO.

The RX FIFO Counter field (RXCTR) in the module's Status Register (SR) indicates the number of valid entries in the RX FIFO. The RXCTR is updated every time the POPR is read or SPI data is copied from the shift register to the RX FIFO.

The POPNXTPTPTR field in the SR points to the RX FIFO entry that is returned when the POPR is read. The POPNXTPTPTR contains the positive offset from RXFR0 in a number of 32-bit registers. For example, POPNXTPTPTR equal to two means that the RXFR2 contains the received SPI data that will be returned when the POPR is read. The POPNXTPTPTR field is incremented every time the POPR is read. The maximum value of the field is equal to the maximum implemented RXFR number and it rolls over after reaching the maximum.

#### **34.5.2.5.1 Filling the RX FIFO**

The RX FIFO is filled with the received SPI data from the shift register. While the RX FIFO is not full, SPI frames from the shift register are transferred to the RX FIFO. Every time an SPI frame is transferred to the RX FIFO, the RX FIFO Counter is incremented by one.

If the RX FIFO and shift register are full and a transfer is initiated, the RFOF bit in the SR is set indicating an overflow condition. Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

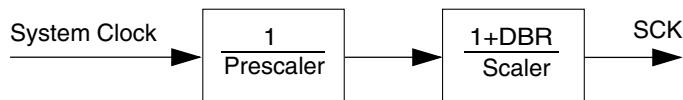
#### **34.5.2.5.2 Draining the RX FIFO**

Host CPU or a DMA can remove (pop) entries from the RX FIFO by reading the module POP RX FIFO Register (POPR). A read of the POPR decrements the RX FIFO Counter by one. Attempts to pop data from an empty RX FIFO are ignored and the RX FIFO Counter remains unchanged. The data, read from the empty RX FIFO, is undetermined.

When the RX FIFO is not empty, the RX FIFO Drain Flag (RFDF) in the SR is set. The RFDF bit is cleared when the RX\_FIFO is empty and the DMA controller indicates that a read from POPR is complete or by writing a 1 to it.

### **34.5.3 Module baud rate and clock delay generation**

The SCK frequency and the delay values for serial transfer are generated by dividing the system clock frequency by a prescaler and a scaler with the option for doubling the baud rate. The following figure shows conceptually how the SCK signal is generated.

**Figure 34-4. Communications clock prescalers and scalers**

### 34.5.3.1 Baud rate generator

The baud rate is the frequency of the SCK. The protocol clock is divided by a prescaler (PBR) and scaler (BR) to produce SCK with the possibility of halving the scaler division. The DBR, PBR, and BR fields in the CTARs select the frequency of SCK by the formula in the BR field description. The following table shows an example of how to compute the baud rate.

**Table 34-10. Baud rate computation example**

<b>f<sub>p</sub></b>	<b>PBR</b>	<b>Prescaler</b>	<b>BR</b>	<b>Scaler</b>	<b>DBR</b>	<b>Baud rate</b>
100 MHz	0b00	2	0b0000	2	0	25 Mb/s
20 MHz	0b00	2	0b0000	2	1	10 Mb/s

#### NOTE

The clock frequencies mentioned in the preceding table are given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

### 34.5.3.2 PCS to SCK Delay (t<sub>csc</sub>)

The PCS to SCK delay is the length of time from assertion of the PCS signal to the first SCK edge. See [Figure 34-5](#) for an illustration of the PCS to SCK delay. The PCSSCK and CSSCK fields in the CTARx registers select the PCS to SCK delay by the formula in the CSSCK field description. The following table shows an example of how to compute the PCS to SCK delay.

**Table 34-11. PCS to SCK delay computation example**

<b>f<sub>p</sub></b>	<b>PCSSCK</b>	<b>Prescaler</b>	<b>CSSCK</b>	<b>Scaler</b>	<b>PCS to SCK Delay</b>
100 MHz	0b01	3	0b0100	32	0.96 µs

**NOTE**

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

**34.5.3.3 After SCK Delay ( $t_{ASC}$ )**

The After SCK Delay is the length of time between the last edge of SCK and the negation of PCS. See [Figure 34-5](#) and [Figure 34-6](#) for illustrations of the After SCK delay. The PASC and ASC fields in the CTARx registers select the After SCK Delay by the formula in the ASC field description. The following table shows an example of how to compute the After SCK delay.

**Table 34-12. After SCK Delay computation example**

$f_P$	PASC	Prescaler	ASC	Scaler	After SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 $\mu$ s

**NOTE**

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

**34.5.3.4 Delay after Transfer ( $t_{DT}$ )**

The Delay after Transfer is the minimum time between negation of the PCS signal for a frame and the assertion of the PCS signal for the next frame. See [Figure 34-5](#) for an illustration of the Delay after Transfer. The PDT and DT fields in the CTARx registers select the Delay after Transfer by the formula in the DT field description. The following table shows an example of how to compute the Delay after Transfer.

**Table 34-13. Delay after Transfer computation example**

$f_P$	PDT	Prescaler	DT	Scaler	Delay after Transfer
100 MHz	0b01	3	0b1110	32768	0.98 ms

**NOTE**

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

When in Non-Continuous Clock mode the  $t_{DT}$  delay is configured according to the equation specified in the CTAR[DT] field description. When in Continuous Clock mode, the delay is fixed at 1 SCK period.

### 34.5.4 Transfer formats

The SPI serial communication is controlled by the Serial Communications Clock (SCK) signal and the PCS signals. The SCK signal provided by the master device synchronizes shifting and sampling of the data on the SIN and SOUT pins. The PCS signals serve as enable signals for the slave devices.

In Master mode, the CPOL and CPHA bits in the Clock and Transfer Attributes Registers (CTARn) select the polarity and phase of the serial clock, SCK.

- CPOL - Selects the idle state polarity of the SCK
- CPHA - Selects if the data on SOUT is valid before or on the first SCK edge

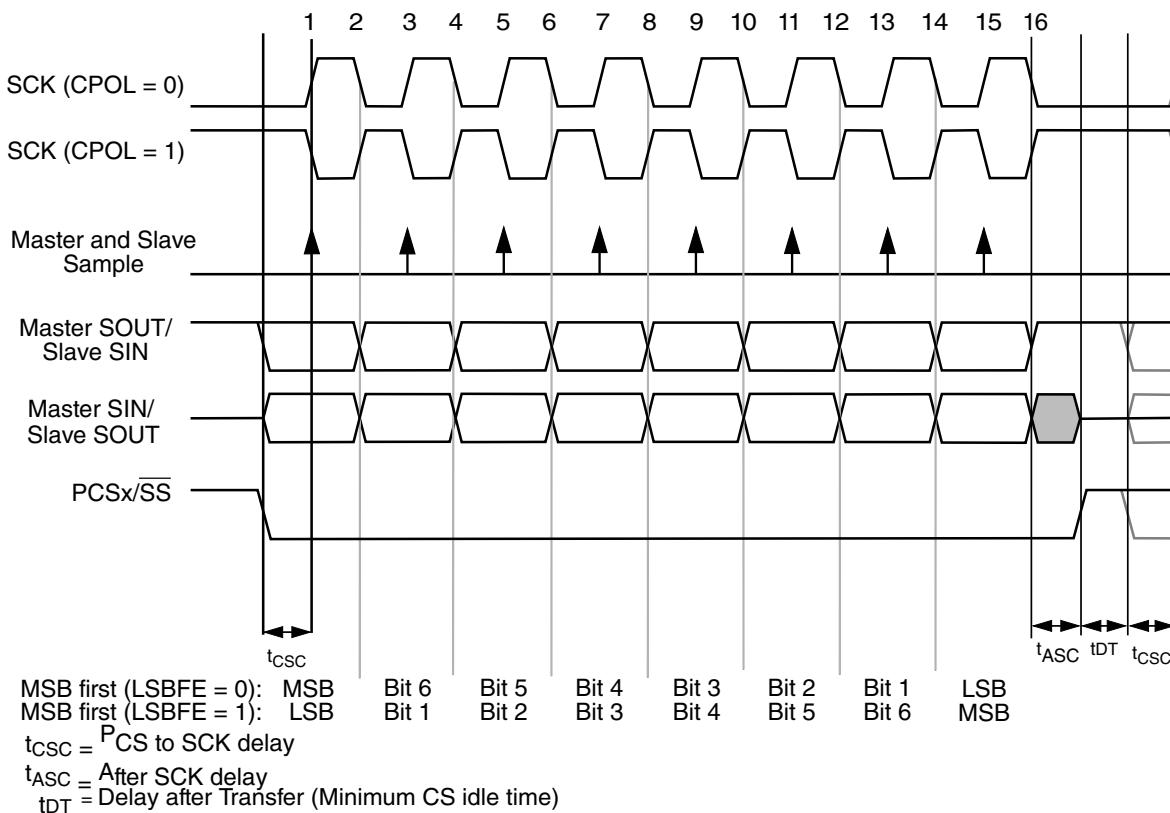
The module supports two different transfer formats:

- Classic SPI with CPHA=0
- Classic SPI with CPHA=1

In the interface configurations, the module provides the option of keeping the PCS signals asserted between frames. See [Continuous Selection Format](#) for details.

#### 34.5.4.1 Classic SPI Transfer Format (CPHA = 0)

The transfer format shown in following figure is used to communicate with peripheral SPI slave devices where the first data bit is available on the first clock edge. In this format, the master and slave sample their SIN pins on the odd-numbered SCK edges and change the data on their SOUT pins on the even-numbered SCK edges.

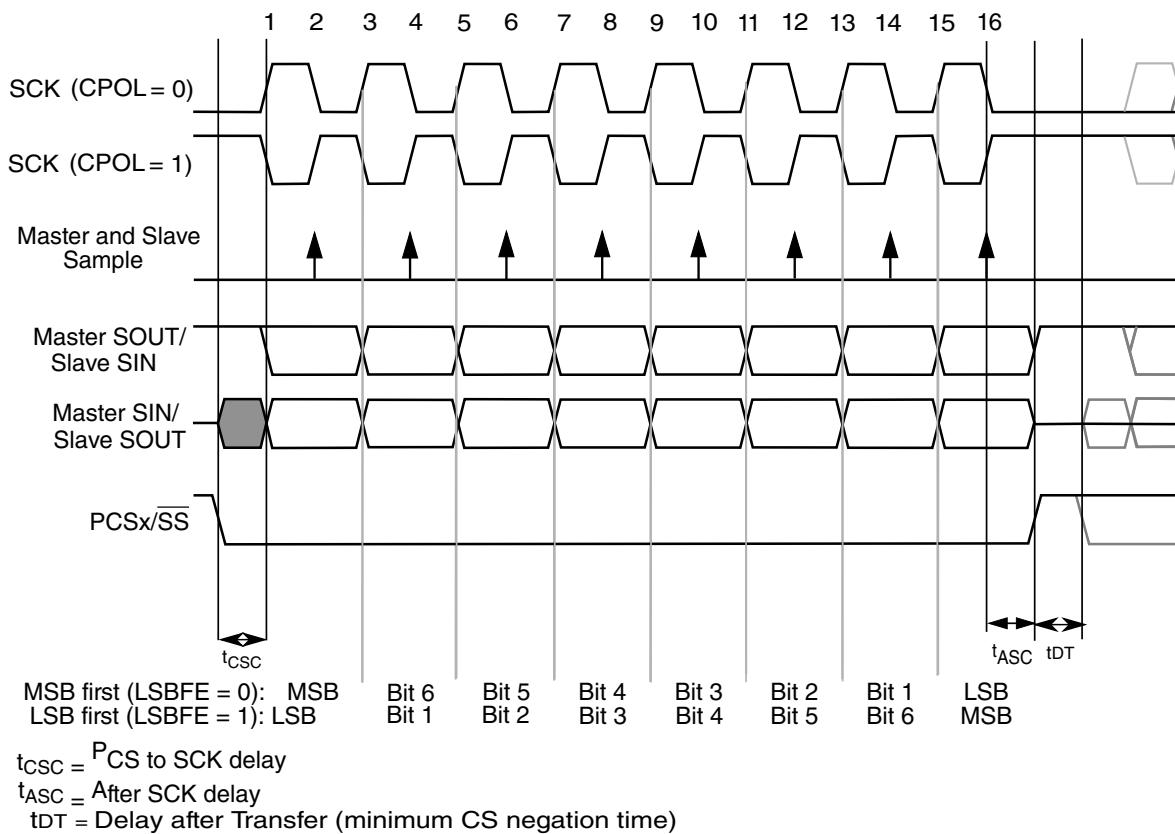


**Figure 34-5. Module transfer timing diagram (CPHA=0, FMSZ=8)**

The master initiates the transfer by placing its first data bit on the SOUT pin and asserting the appropriate peripheral chip select signals to the slave device. The slave responds by placing its first data bit on its SOUT pin. After the  $t_{CSC}$  delay elapses, the master outputs the first edge of SCK. The master and slave devices use this edge to sample the first input data bit on their serial data input signals. At the second edge of the SCK, the master and slave devices place their second data bit on their serial data output signals. For the rest of the frame the master and the slave sample their SIN pins on the odd-numbered clock edges and changes the data on their SOUT pins on the even-numbered clock edges. After the last clock edge occurs, a delay of  $t_{ASC}$  is inserted before the master negates the PCS signals. A delay of  $t_{DT}$  is inserted before a new frame transfer can be initiated by the master.

### 34.5.4.2 Classic SPI Transfer Format (CPHA = 1)

This transfer format shown in the following figure is used to communicate with peripheral SPI slave devices that require the first SCK edge before the first data bit becomes available on the slave SOUT pin. In this format, the master and slave devices change the data on their SOUT pins on the odd-numbered SCK edges and sample the data on their SIN pins on the even-numbered SCK edges.



**Figure 34-6. Module transfer timing diagram (CPHA=1, FMSZ=8)**

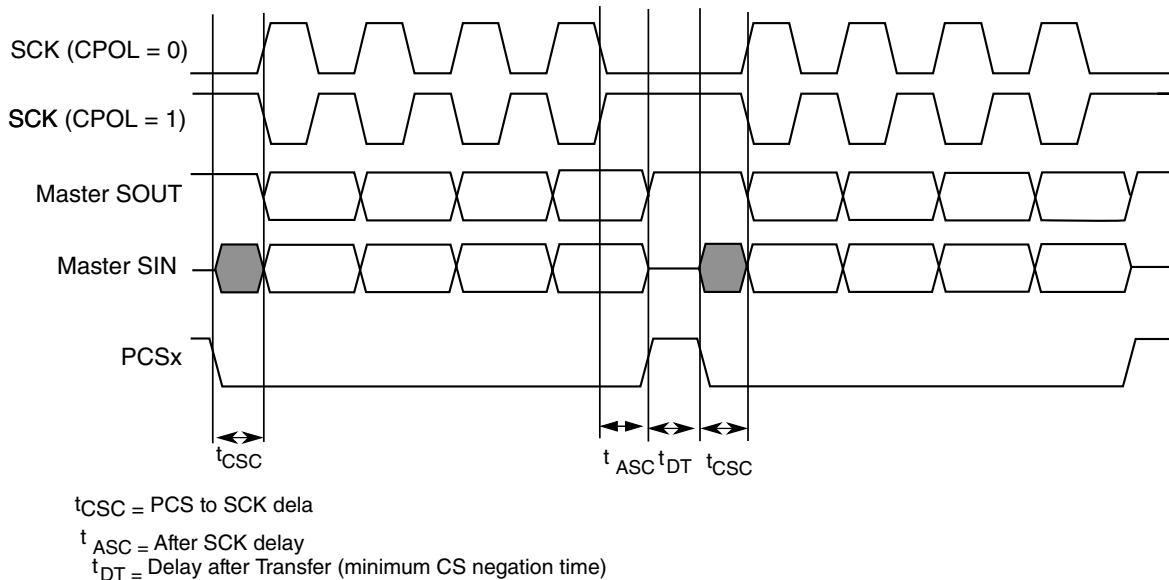
The master initiates the transfer by asserting the PCS signal to the slave. After the  $t_{CSC}$  delay has elapsed, the master generates the first SCK edge and at the same time places valid data on the master SOUT pin. The slave responds to the first SCK edge by placing its first data bit on its slave SOUT pin.

At the second edge of the SCK the master and slave sample their SIN pins. For the rest of the frame the master and the slave change the data on their SOUT pins on the odd-numbered clock edges and sample their SIN pins on the even-numbered clock edges. After the last clock edge occurs, a delay of  $t_{ASC}$  is inserted before the master negates the PCS signal. A delay of  $t_{DT}$  is inserted before a new frame transfer can be initiated by the master.

### 34.5.4.3 Continuous Selection Format

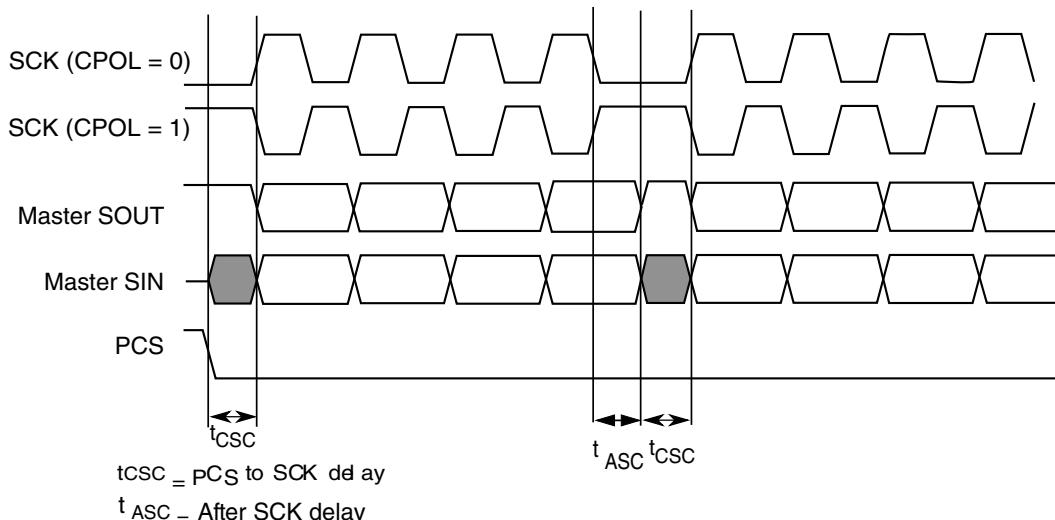
Some peripherals must be deselected between every transfer. Other peripherals must remain selected between several sequential serial transfers. The Continuous Selection Format provides the flexibility to handle the following case. The Continuous Selection Format is enabled for the SPI configuration by setting the CONT bit in the SPI command.

When the CONT bit = 0, the module drives the asserted Chip Select signals to their idle states in between frames. The idle states of the Chip Select signals are selected by the PCSISn bits in the MCR. The following timing diagram is for two four-bit transfers with CPHA = 1 and CONT = 0.



**Figure 34-7. Example of non-continuous format (CPHA=1, CONT=0)**

When the CONT bit = 1, the PCS signal remains asserted for the duration of the two transfers. The Delay between Transfers ( $t_{DT}$ ) is not inserted between the transfers. The following figure shows the timing diagram for two four-bit transfers with CPHA = 1 and CONT = 1.



**Figure 34-8. Example of continuous transfer (CPHA=1, CONT=1)**

When using the module with continuous selection follow these rules:

- All transmit commands must have the same PCSn bits programming.
- The CTARs, selected by transmit commands, must be programmed with the same transfer attributes. Only FMSZ field can be programmed differently in these CTARs.
- When transmitting multiple frames in this mode, the user software must ensure that the last frame has the PUSHR[CONT] bit deasserted in Master mode and the user software must provide sufficient frames in the TX\_FIFO to be sent out in Slave mode and the master deasserts the PCSn at end of transmission of the last frame.
- PUSHR[CONT] must be deasserted before asserting MCR[HALT] in master mode. This will make sure that the PCSn signals are deasserted. Asserting MCR[HALT] during continuous transfer will cause the PCSn signals to remain asserted and hence Slave Device cannot transition from Running to Stopped state.

**NOTE**

User must fill the TX FIFO with the number of entries that will be concatenated together under one PCS assertion for both master before the TX FIFO becomes empty.

#### **34.5.4.4 Fast Continuous Selection Format**

The Fast Continuous Selection Format functions similar to [Continuous Selection Format](#) except that the inter command delays,  $t_{ASC}$  and  $t_{CSC}$ , can be masked out and are not inserted by the hardware.

**NOTE**

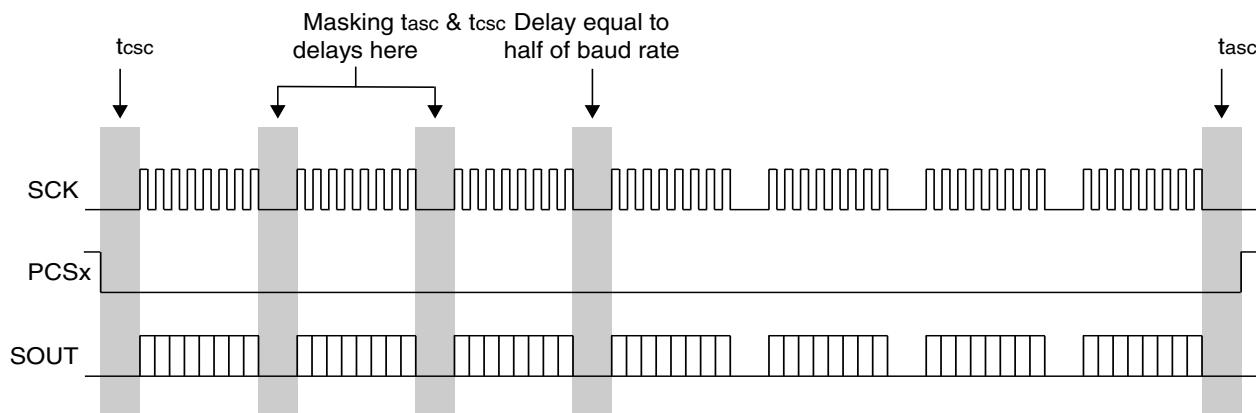
The Fast Continuous Selection Format is available in the SPI configuration only and when Continuous Serial Communication Clock mode is disabled. Masking of delays is not allowed if the transfer is non-continuous.

The Fast Continuous Selection Format is enabled by writing ‘1’ into FCPCS bit of the MCR register. When this bit is asserted, MASC and MCSC bits of the PUSHR register perform the function of mask bits for the transmit frame. These bits individually mask the  $t_{ASC}$  and  $t_{CSC}$  delays as programmed by the user software. A normal Continuous Selection Format has these two delays for each frame that is transmitted with the CONT bit asserted. In order to avoid these delays and to speed up the transfer process, the software can simply mask these delays while programming the command in the PUSHR register.

While masking the delays, the software must follow the following masking rules, else correct operation is not guaranteed.

- MASC bit masks the “After SCK” delay for the current frame.
- MCSC bit masks the “PCS to SCK” delay for the next frame.
- “After SCK” ( $t_{ASC}$ ) delay must not be masked when the current frame is the last frame in the continuous selection format.
- The “PCS to SCK” delay for the first frame in the continuous selection format cannot be masked.
- Masking of only  $t_{ASC}$  is not allowed. If  $t_{ASC}$  is masked then  $t_{CSC}$  must be masked too.
- Masking of both  $t_{ASC}$  and  $t_{CSC}$  delays is allowed. In this case, the delay between two frames is equal to half the baud rate set by the user software.
- Masking of only  $t_{CSC}$  is allowed. In this case, the delay between two frames is equal to the  $t_{ASC}$  time and thus the user software must ensure that the  $t_{ASC}$  time is greater than the baud rate.
- The user software must not mask these delays if the continuous selection format is not used and MCR[FCPCS] is asserted.
- Rules applicable to the Continuous Selection Format are applicable here too.

The following figure shows the timing for a Fast Continuous Selection Format transfer. Here seven frames are transferred with both  $t_{ASC}$  and  $t_{CSC}$  delays masked except for the last frame that terminated the transfer. The last frame has  $t_{ASC}$  delay at its end.



**Figure 34-9. Example of Fast Continuous Selection Format**

In case any chip select is to be changed, then the fast continuous selection format should be terminated and then the chips selects should change and appropriate delays must be introduced.

### 34.5.5 Continuous Serial Communications Clock

The module provides the option of generating a Continuous SCK signal for slave peripherals that require a continuous clock.

Continuous SCK is enabled by setting the CONT\_SCKE bit in the MCR. Enabling this bit generates the Continuous SCK only if MCR[HALT] bit is low. Continuous SCK is valid in all configurations.

Continuous SCK is only supported for CPHA=1. Clearing CPHA is ignored if the CONT\_SCKE bit is set. .

Clock and transfer attributes for the Continuous SCK mode are set according to the following rules:

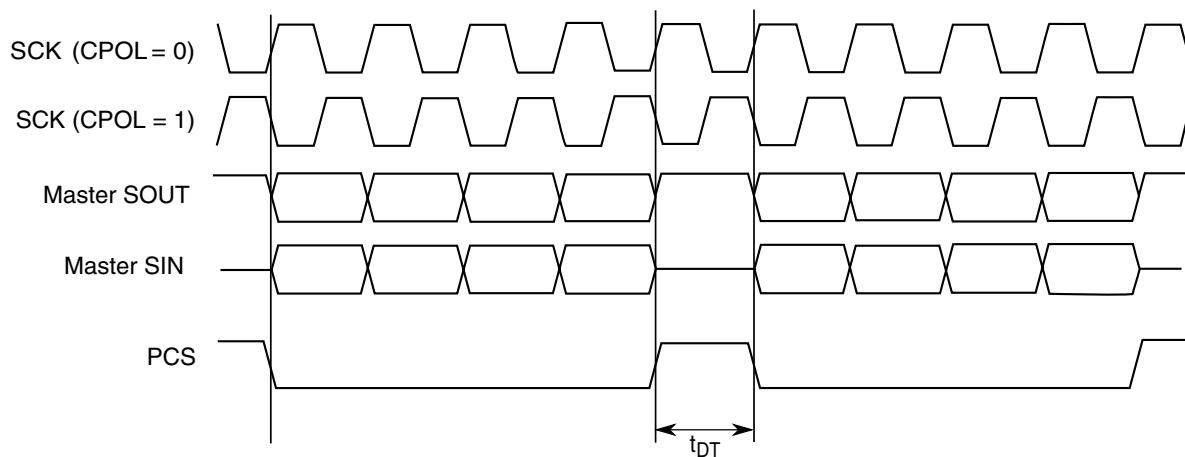
- When the module is in SPI configuration, CTAR0 is used initially. At the start of each SPI frame transfer, the CTAR specified by the CTAS for the frame is used.
- In all configurations, the currently selected CTAR remains in use until the start of a frame with a different CTAR specified, or the Continuous SCK mode is terminated.

It is recommended to keep the baud rate the same while using the Continuous SCK. Switching clock polarity between frames while using Continuous SCK can cause errors in the transfer. Continuous SCK operation is not guaranteed if the module is put into the External Stop mode or Module Disable mode.

Enabling Continuous SCK disables the PCS to SCK delay and the Delay after Transfer ( $t_{DT}$ ) is fixed to one SCK cycle. The following figure is the timing diagram for Continuous SCK format with Continuous Selection disabled.

#### NOTE

In Continuous SCK mode, for the SPI transfer CTAR0 should always be used, and the TX FIFO must be cleared using the MCR[CLR\_TXF] field before initiating transfer.

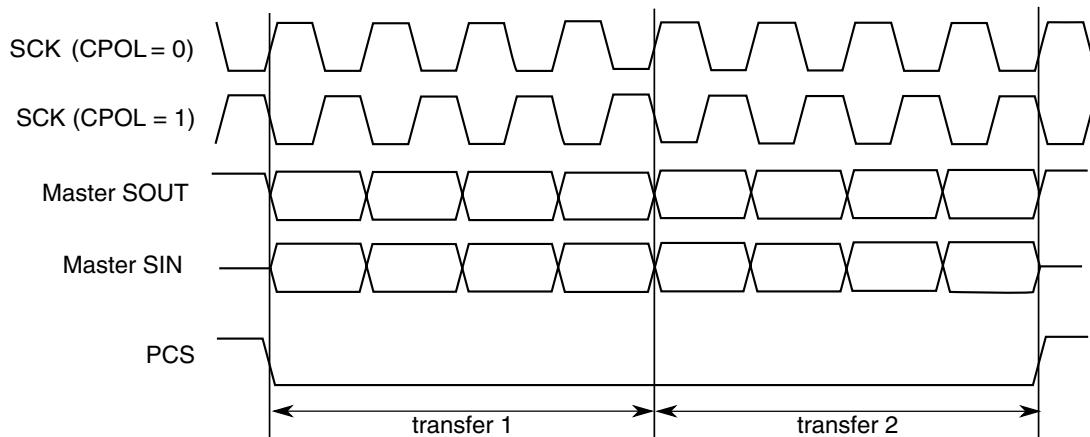


**Figure 34-10. Continuous SCK Timing Diagram (CONT=0)**

If the CONT bit in the TX FIFO entry is set, PCS remains asserted between the transfers. Under certain conditions, SCK can continue with PCS asserted, but with no data being shifted out of SOUT, that is, SOUT pulled high. This can cause the slave to receive incorrect data. Those conditions include:

- Continuous SCK with CONT bit set, but no data in the TX FIFO.
- Continuous SCK with CONT bit set and entering Stopped state (refer to [Start and Stop of module transfers](#)).
- Continuous SCK with CONT bit set and entering Stop mode or Module Disable mode.

The following figure shows timing diagram for Continuous SCK format with Continuous Selection enabled.



**Figure 34-11. Continuous SCK timing diagram (CONT=1)**

## 34.5.6 Parity Generation and Check

The module can generate and check parity in the serial frame. The parity bit replaces the last transmitted bit in the frame. The parity is calculated for all transmitted data bits in frame, not including the last data bit that would be transmitted. The parity generation/control is done on frame basis. The registers field setting frame size defines the total number of bits in the frame, including the parity bit. Thus, to transmit/receive the same number of data bits with parity check, increase the frame size by one versus the same data size frame without the parity check.

Parity can be selected as odd or even. Parity Errors in the received frame set Parity Error flags in the Status register. The Parity Error Interrupt Requests are generated if enabled. The module can be programmed to stop frame transmission in case of a frame reception with parity error.

### 34.5.6.1 Parity for SPI Frames

When the module is in the master mode the parity generation is controlled by PE and PP bits of the CMD FIFO entries (PUSHR). Setting the PE bit enables parity generation for transmitted SPI frames and parity check for received frames. PP bit defines polarity of the parity bit.

When continuous PCS selection is used to transmit SPI data, two parity generation scenarios are available:

- Generate/check parity for the whole frame
- Generate/check parity for each sub-frame separately.

To generate/check parity for the whole frame set PE bit only in the last command/TX FIFO entry, forming this frame (with the PUSHR register).

To generate/check parity for each sub-frame set PE bit in each command/TX FIFO entry, forming this frame.

If the parity error occurs for received SPI frame, the SR[SPEF] bit is set. If MCR[PES] bit is set, the module stops SPI frames transmission. To resume SPI operation clear the SR[SPEF] or the MCR[PES] bits.

### 34.5.7 Interrupts/DMA requests

The module has several conditions that can generate only interrupt requests and two conditions that can generate interrupt or DMA requests. The following table lists these conditions.

**Table 34-14. Interrupt and DMA request conditions**

Condition	Flag	Interrupt	DMA
End of Queue (EOQ)	EOQF	Yes	-
TX FIFO Fill	TFFF	Yes	Yes
CMD FIFO Fill	CMDFFF	Yes	Yes
TX FIFO Invalid Write	TFIWF	Yes	-
Transfer Complete	TCF	Yes	-
CMD Transfer Complete	CMDTCF	Yes	-
TX FIFO Underflow	TFUF	Yes	-
RX FIFO Drain	RFDF	Yes	Yes
RX FIFO Overflow	RFOF	Yes	-
SPI Parity Error	SPEF	Yes	-

Each condition has a flag bit in the module Status Register (SR) and a Request Enable bit in the DMA/Interrupt Request Select and Enable Register (RSER). Certain flags (as shown in above table) generate interrupt requests or DMA requests depending on configuration of RSER register.

The module also provides a global interrupt request line, which is asserted when any of individual interrupt requests lines is asserted.

#### 34.5.7.1 End Of Queue interrupt request

The End Of Queue (EOQ) interrupt request indicates that the end of a transmit queue is reached. The module generates the interrupt request when EOQ interrupt requests are enabled (RSER[EOQF\_RE]) and the EOQ bit in the executing SPI command is 1.

When Extended SPI mode is enabled (MCR[XSPI]) and the EOQ bit in the executing SPI command is 1, the module generates the EOQ interrupt request when the last bit of the last data frame in the command cycle has been transmitted.

When Extended SPI mode is disabled (MCR[XSPI]), the module generates the interrupt request when the last bit of the SPI frame with EOQ bit set is transmitted.

### 34.5.7.2 Transmit FIFO Fill Interrupt or DMA Request

The Transmit FIFO Fill Request indicates that the TX FIFO is not full. The Transmit FIFO Fill Request is generated when the number of entries in the TX FIFO is less than the maximum number of possible entries, and the TFFF\_RE bit in the RSER is set. The TFFF\_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated.

#### NOTE

TFFF flag clears automatically when DMA is used to fill TX FIFO. Configure the DMA to fill only one FIFO location per transfer.

To clear TFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill TX FIFO:

1. Wait until TFFF = 1.
2. Write data to PUSHR using CPU.
3. Clear TFFF by writing a 1 to its location. If TX FIFO is not full, this flag will not clear.

### 34.5.7.3 Command FIFO Fill Interrupt or DMA Request

The Command FIFO Fill Request indicates that the CMD FIFO is not full. The Command FIFO Fill Request is generated when the number of entries in the CMD FIFO is less than the maximum number of possible entries, and the CMDFFF\_RE bit in the RSER is set. The CMDFFF\_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated.

This Request is useful when MCR[XSPI] is enabled, since TX FIFO and CMD FIFO can be filled independantly. If MCR[XSPI] is disabled, then ‘TX FIFO Fill Interrupt or DMA Request’ will suffice to fill both FIFO’s since both FIFO’s must be filled simultaneously.

#### NOTE

CMDFFF flag clears automatically when DMA is used to fill CMD FIFO. Configure the DMA to fill only one FIFO location per transfer.

To clear CMDFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill CMDFIFO:

1. Wait until CMDFFF = 1
2. Write data to PUSHR using CPU.

3. Clear CMDFFF by writing a 1 to its location. If CMD FIFO is not full, this flag will not clear.

#### 34.5.7.4 Transmit FIFO Invalid Write Interrupt Request

The Transmit FIFO Invalid Write Request is valid only when MCR[XSPI] is enabled. This Request indicates that Data exists in the TX FIFO while the CMD FIFO is empty. Since no Command Fields are associated with the Data present in TX FIFO, this data is considered invalid until a Command Entry becomes available. The Transmit FIFO Invalid Write Request is generated for the above condition when TFIWF\_RE bit is set in the RSER.

#### 34.5.7.5 Transfer Complete Interrupt Request

The Transfer Complete Request indicates the end of the transfer of a serial frame. The Transfer Complete Request is generated at the end of each frame transfer when the TCF\_RE bit is set in the RSER.

#### 34.5.7.6 Command Transfer Complete Interrupt Request

The Command Transfer Complete Request indicates the end of transfer of the last SPI frame in a Command Cycle. The Transfer Complete Request is generated for the above condition when the CMDTCF\_RE bit is set in the RSER.

#### 34.5.7.7 Receive FIFO Drain Interrupt or DMA Request

The Receive FIFO Drain Request indicates that the RX FIFO is not empty. The Receive FIFO Drain Request is generated when the number of entries in the RX FIFO is not zero, and the RFDF\_RE bit in the RSER is set. The RFDF\_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated. Configure the DMA to drain only one FIFO location per transfer.

### 34.5.7.8 Receive FIFO Overflow Interrupt Request

The Receive FIFO Overflow Request indicates that an overflow condition in the RX FIFO has occurred. A Receive FIFO Overflow request is generated when RX FIFO and shift register are full and a transfer is initiated. The RFOF\_RE bit in the RSER must be set for the interrupt request to be generated.

Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

### 34.5.7.9 SPI Frame Parity Error Interrupt Request

The SPI Frame Parity Error Flag indicates that a SPI frame with parity error had been received. The SPEF\_RE bit in the RSER must be set for the interrupt request to be generated.

## 34.5.8 Power saving features

The module supports following power-saving strategies:

- External Stop mode
- Module Disable mode – Clock gating of non-memory mapped logic

### 34.5.8.1 Stop mode (External Stop mode)

This module supports the Stop mode protocol. When a request is made to enter External Stop mode, the module acknowledges the request . If a serial transfer is in progress, then this module waits until it reaches the frame boundary before it is ready to have its clocks shut off . While the clocks are shut off, this module's memory-mapped logic is not accessible. This also puts the module in STOPPED state. The SR[TXRXS] bit is cleared to indicate STOPPED state. The states of the interrupt and DMA request signals cannot be changed while in External Stop mode.

### 34.5.8.2 Module Disable mode

Module Disable mode is a block-specific mode that the module can enter to save power. Host CPU can initiate the Module Disable mode by setting the MDIS bit in the MCR. The Module Disable mode can also be initiated by hardware.

When the MDIS bit is set, the module negates the Clock Enable signal at the next frame boundary. Once the Clock Enable signal is negated, it is said to have entered Module Disable Mode. This also puts the module in STOPPED state. The SR[TXRXS] bit is cleared to indicate STOPPED state. If implemented, the Clock Enable signal can stop the clock to the non-memory mapped logic. When Clock Enable is negated, the module is in a dormant state, but the memory mapped registers are still accessible. Certain read or write operations have a different effect when the module is in the Module Disable mode. Reading the RX FIFO Pop Register does not change the state of the RX FIFO. Similarly, writing to the PUSHR Register does not change the state of the TX FIFO or CMD FIFO. Clearing either of the FIFOs has no effect in the Module Disable mode. Changes to the DIS\_TXF and DIS\_RXF fields of the MCR have no effect in the Module Disable mode. In the Module Disable mode, all status bits and register flags in the module return the correct values when read, but writing to them has no effect. Writing to the TCR during Module Disable mode has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

## 34.6 Initialization/application information

This section describes how to initialize the module.

### 34.6.1 How to manage queues

The queues are not part of the module, but it includes features in support of queue management. Queues are primarily supported in SPI configuration.

1. When module executes last command word from a queue, the EOQ bit in the command word is set to indicate it that this is the last entry in the queue.
2. At the end of the transfer, corresponding to the command word with EOQ set is sampled, the EOQ flag (EOQF) in the SR is set.
3. The setting of the EOQF flag disables serial transmission and reception of data, putting the module in the Stopped state. The TXRXS bit is cleared to indicate the Stopped state.
4. The DMA can continue to fill TX FIFO until it is full or step 5 occurs.

5. Disable DMA transfers by disabling the DMA enable request for the DMA channel assigned to TX FIFO (and CMD FIFO) and RX FIFO. This is done by clearing the corresponding DMA enable request bits in the DMA Controller.
6. Ensure all received data in RX FIFO has been transferred to memory receive queue by reading the RXCNT in SR or by checking RFDF in the SR after each read operation of the POPR.
7. Modify DMA descriptor of TX and RX channels for new queues
8. Flush TX FIFO (and CMD FIFO) by writing a 1 to the CLR\_TXF bit in the MCR. Flush RX FIFO by writing a '1' to the CLR\_RXF bit in the MCR.
9. Clear transfer count either by setting CTCNT bit in the command word of the first entry in the new queue or via CPU writing directly to SPI\_TCNT field in the TCR.
10. Enable DMA channel by enabling the DMA enable request for the DMA channel assigned to the module TX FIFO, (and CMD FIFO) and RX FIFO by setting the corresponding DMA set enable request bit.
11. Enable serial transmission and serial reception of data by clearing the EOQF bit.

### **34.6.2 Initializing Module in Master Mode**

Once the appropriate mode in MCR[MSTR] is configured, the module is enabled by clearing MCR[HALT]. It should be ensured that module Slave is enabled before enabling it's Master. This ensures the Slave is ready to be communicated with, before Master initializes communication.

### **34.6.3 Baud rate settings**

The following table shows the baud rate that is generated based on the combination of the baud rate prescaler PBR and the baud rate scaler BR in the CTARs. The values calculated assume a 100 MHz protocol frequency and the double baud rate DBR bit is cleared.

**Table 34-15. Baud rate values (bps)**

		Baud rate divider prescaler values			
		2	3	5	7
Baud Rate Scaler Values	2	25.0M	16.7M	10.0M	7.14M
	4	12.5M	8.33M	5.00M	3.57M
	6	8.33M	5.56M	3.33M	2.38M
	8	6.25M	4.17M	2.50M	1.79M
	16	3.12M	2.08M	1.25M	893k
	32	1.56M	1.04M	625k	446k
	64	781k	521k	312k	223k
	128	391k	260k	156k	112k
	256	195k	130k	78.1k	55.8k
	512	97.7k	65.1k	39.1k	27.9k
	1024	48.8k	32.6k	19.5k	14.0k
	2048	24.4k	16.3k	9.77k	6.98k
	4096	12.2k	8.14k	4.88k	3.49k
	8192	6.10k	4.07k	2.44k	1.74k
	16384	3.05k	2.04k	1.22k	872
	32768	1.53k	1.02k	610	436

### 34.6.4 Delay settings

The following table shows the values for the Delay after Transfer ( $t_{DT}$ ) and CS to SCK Delay ( $T_{CSC}$ ) that can be generated based on the prescaler values and the scaler values set in the CTARs. The values calculated assume a 100 MHz protocol frequency.

#### NOTE

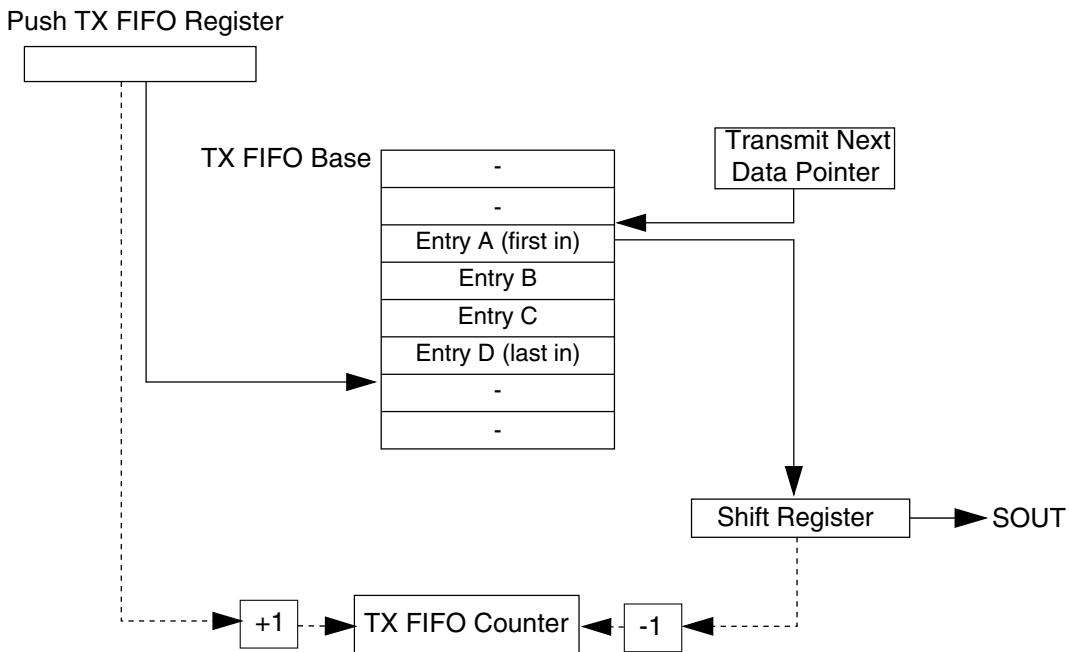
The clock frequency mentioned above is given as an example in this chapter. See the clocking chapter for the frequency used to drive this module in the device.

**Table 34-16. Delay values**

		Delay prescaler values			
		1	3	5	7
Delay scaler values	2	20.0 ns	60.0 ns	100.0 ns	140.0 ns
	4	40.0 ns	120.0 ns	200.0 ns	280.0 ns
	8	80.0 ns	240.0 ns	400.0 ns	560.0 ns
	16	160.0 ns	480.0 ns	800.0 ns	1.1 µs
	32	320.0 ns	960.0 ns	1.6 µs	2.2 µs
	64	640.0 ns	1.9 µs	3.2 µs	4.5 µs
	128	1.3 µs	3.8 µs	6.4 µs	9.0 µs
	256	2.6 µs	7.7 µs	12.8 µs	17.9 µs
	512	5.1 µs	15.4 µs	25.6 µs	35.8 µs
	1024	10.2 µs	30.7 µs	51.2 µs	71.7 µs
	2048	20.5 µs	61.4 µs	102.4 µs	143.4 µs
	4096	41.0 µs	122.9 µs	204.8 µs	286.7 µs
	8192	81.9 µs	245.8 µs	409.6 µs	573.4 µs
	16384	163.8 µs	491.5 µs	819.2 µs	1.1 ms
	32768	327.7 µs	983.0 µs	1.6 ms	2.3 ms
	65536	655.4 µs	2.0 ms	3.3 ms	4.6 ms

### 34.6.5 Calculation of FIFO pointer addresses

Complete visibility of the FIFO contents is available through the FIFO registers, and valid entries can be identified through a memory-mapped pointer and counter for each FIFO. The pointer to the first-in entry in each FIFO is memory mapped. For the TX FIFO the first-in pointer is the Transmit Next Pointer (TXNXTPTR). For the CMD FIFO the first-in pointer is the Command Next Pointer (CMDNXTPTR). For the RX FIFO the first-in pointer is the Pop Next Pointer (POPNXTPTR). The following figure illustrates the concept of first-in and last-in FIFO entries along with the FIFO Counter. The TX FIFO is chosen for the illustration, but the concepts carry over. See [Transmit First In First Out \(TX FIFO\) buffering mechanism](#), [Command First In First Out \(CMD FIFO\) Buffering Mechanism](#) and [Receive First In First Out \(RX FIFO\) buffering mechanism](#) for details on the FIFO operation.



**Figure 34-12. TX FIFO pointers and counter**

### 34.6.5.1 Address Calculation for the First-in Entry and Last-in Entry in the TX FIFO

The memory address of the first-in entry in the TX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{TXFIFOBase} + (4 \times \text{TXNXTPTR})$$

The memory address of the last-in entry in the TX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{TXFIFOBase} + 4 \times (\text{TXCTR} + \text{TXNXTPTR} - 1) \bmod (\text{TXFIFOdepth})$$

TX FIFO Base - Base address of TX FIFO

TXCTR - TX FIFO Counter

TXNXTPTR - Transmit Next Pointer

TX FIFO Depth - Transmit FIFO depth, implementation specific

### 34.6.5.2 Address Calculation for the First-in Entry and Last-in Entry in the CMD FIFO

The memory address of the first-in entry in the CMD FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{TXFIFOBase} + (4 \times \text{TXNXTPTR})$$

The memory address of the last-in entry in the CMD FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{TXFIFOBase} + 4 \times (\text{TXCTR} + \text{TXNXTPTR} - 1) \bmod (\text{TXFIFOdepth})$$

CMD FIFO Base - Base address of CMD FIFO

CMDCTR - CMD FIFO Counter

CMDNXTPTR - Command Next Pointer

CMD FIFO Depth - Command FIFO depth, implementation specific

### 34.6.5.3 Address Calculation for the First-in Entry and Last-in Entry in the RX FIFO

The memory address of the first-in entry in the RX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{RX FIFOBase} + (4 \times \text{POPNXTPTR})$$

The memory address of the last-in entry in the RX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{RX FIFO Base} + 4 \times (\text{RXCTR} + \text{POPNXTPTR} - 1) \bmod (\text{RXFIFOdepth})$$

RX FIFO Base - Base address of RX FIFO

RXCTR - RX FIFO counter

POPNXTPTR - Pop Next Pointer

RX FIFO Depth - Receive FIFO depth, implementation specific

# Chapter 35

## Thermal Monitoring Unit (TMU)

### 35.1 The TMU module as implemented on the chip

This section provides details about how the TMU module is implemented on the chip.

#### NOTE

- The maximum operating temperature of the chip should not exceed that specified in the data sheet. For more information, refer the data sheet.
- Accuracy within  $\pm 3^{\circ}\text{C}$

#### 35.1.1 Local temperature sensor placement

The table below shows the placement of the local temperature sensor:

**Table 35-1. Local temperature sensor placement in the chip**

Temperature sensor ID	Placement
0	Near DDR controller
1	Near SerDes
2	Near Frame manager
3	Near Arm A53 core
4	SEC
5-15	Reserved

### 35.1.2 Initialization Information

The TMU calculates temperature by reading one of the temperature sensor sites on the chip. The temperature is calculated from a calibration table, which is programmed either by the pre-boot loader or initialization of calibration table through software. Failure to properly initialize the calibration table may lead to an undefined behavior. Calibration determines specific sensor reading translated in degrees Celsius as shown in the following tables.

**Table 35-2. Temperature calibration points**

Register Name	Value
<b>TTR0CR, 12 points at 0°C</b>	
Point 0 at 0°C	
TTCFGR	0x0000_0000
TSCFGR	0x0000_0023
Point 1 at 4°C	
TTCFGR	0x0000_0001
TSCFGR	0x0000_002A
Point 2 at 8°C	
TTCFGR	0x0000_0002
TSCFGR	0x0000_0031
Point 3 at 12°C	
TTCFGR	0x0000_0003
TSCFGR	0x0000_0037
Point 4 at 16°C	
TTCFGR	0x0000_0004
TSCFGR	0x0000_003E
Point 5 at 20°C	
TTCFGR	0x0000_0005
TSCFGR	0x0000_0044
Point 6 at 24°C	
TTCFGR	0x0000_0006
TSCFGR	0x0000_004B
Point 7 at 28°C	
TTCFGR	0x0000_0007
TSCFGR	0x0000_0051
Point 8 at 32°C	
TTCFGR	0x0000_0008
TSCFGR	0x0000_0058
Point 9 at 36°C	
TTCFGR	0x0000_0009
TSCFGR	0x0000_005E

*Table continues on the next page...*

**Table 35-2. Temperature calibration points (continued)**

Register Name	Value
Point 10 at 40°C	
TTCFGR	0x0000_000A
TSCFGR	0x0000_0065
Point 11 at 44°C	
TTCFGR	0x0000_000B
TSCFGR	0x0000_006B
<b>TTR1CR, 10 points at 42°C</b>	
Point 0 at 42°C	
TTCFGR	0x0001_0000
TSCFGR	0x0000_0023
Point 1 at 46°C	
TTCFGR	0x0001_0001
TSCFGR	0x0000_002B
Point 2 at 50°C	
TTCFGR	0x0001_0002
TSCFGR	0x0000_0033
Point 3 at 54°C	
TTCFGR	0x0001_0003
TSCFGR	0x0000_003B
Point 4 at 58°C	
TTCFGR	0x0001_0004
TSCFGR	0x0000_0043
Point 5 at 62°C	
TTCFGR	0x0001_0005
TSCFGR	0x0000_004B
Point 6 at 66°C	
TTCFGR	0x0001_0006
TSCFGR	0x0000_0054
Point 7 at 70°C	
TTCFGR	0x0001_0007
TSCFGR	0x0000_005C
Point 8 at 74°C	
TTCFGR	0x0001_0008
TSCFGR	0x0000_0064
Point 9 at 78°C	
TTCFGR	0x0001_0009
TSCFGR	0x0000_006C
<b>TTR2CR, 7 points at 76°C</b>	
Point 0 at 76°C	

*Table continues on the next page...*

**Table 35-2. Temperature calibration points (continued)**

Register Name	Value
TTCFGR	0x0002_0000
TSCFGR	0x0000_0021
Point 1 at 80°C	
TTCFGR	0x0002_0001
TSCFGR	0x0000_002C
Point 2 at 84°C	
TTCFGR	0x0002_0002
TSCFGR	0x0000_0036
Point 3 at 88°C	
TTCFGR	0x0002_0003
TSCFGR	0x0000_0040
Point 4 at 92°C	
TTCFGR	0x0002_0004
TSCFGR	0x0000_004B
Point 5 at 96°C	
TTCFGR	0x0002_0005
TSCFGR	0x0000_0055
Point 6 at 100°C	
TTCFGR	0x0002_0006
TSCFGR	0x0000_005F
<b>TTR3CR, 8 points at 98°C</b>	
Point 0 at 98°C	
TTCFGR	0x0003_0000
TSCFGR	0x0000_0013
Point 1 at 102°C	
TTCFGR	0x0003_0001
TSCFGR	0x0000_001D
Point 2 at 106°C	
TTCFGR	0x0003_0002
TSCFGR	0x0000_0028
Point 3 at 110°C	
TTCFGR	0x0003_0003
TSCFGR	0x0000_0032
Point 4 at 114°C	
TTCFGR	0x0003_0004
TSCFGR	0x0000_003D
Point 5 at 118°C	
TTCFGR	0x003_0005
TSCFGR	0x0000_0047

*Table continues on the next page...*

**Table 35-2. Temperature calibration points (continued)**

Register Name	Value
Point 6 at 122°C	
TTCFGR	0x0003_0006
TSCFGR	0x0000_0052
Point 7 at 126°C	
TTCFGR	0x0003_0007
TSCFGR	0x0000_005C

The process for programming the calibration table is described below:

1. Disable the monitoring mode, TMR[ME]=0 (default).
2. Program TMU\_TTR0CR=0x000B0000, TMU\_TTR1CR=0x0009002A, TMU\_TTR2CR=0x0006004C and TMU\_TTR3CR=0x00070062
3. Write the temperature configuration register (TTCFGR).
4. Write the sensor configuration register (TSCFGR).
5. Repeat steps 3-4 for all temperatures as defined in the above table.

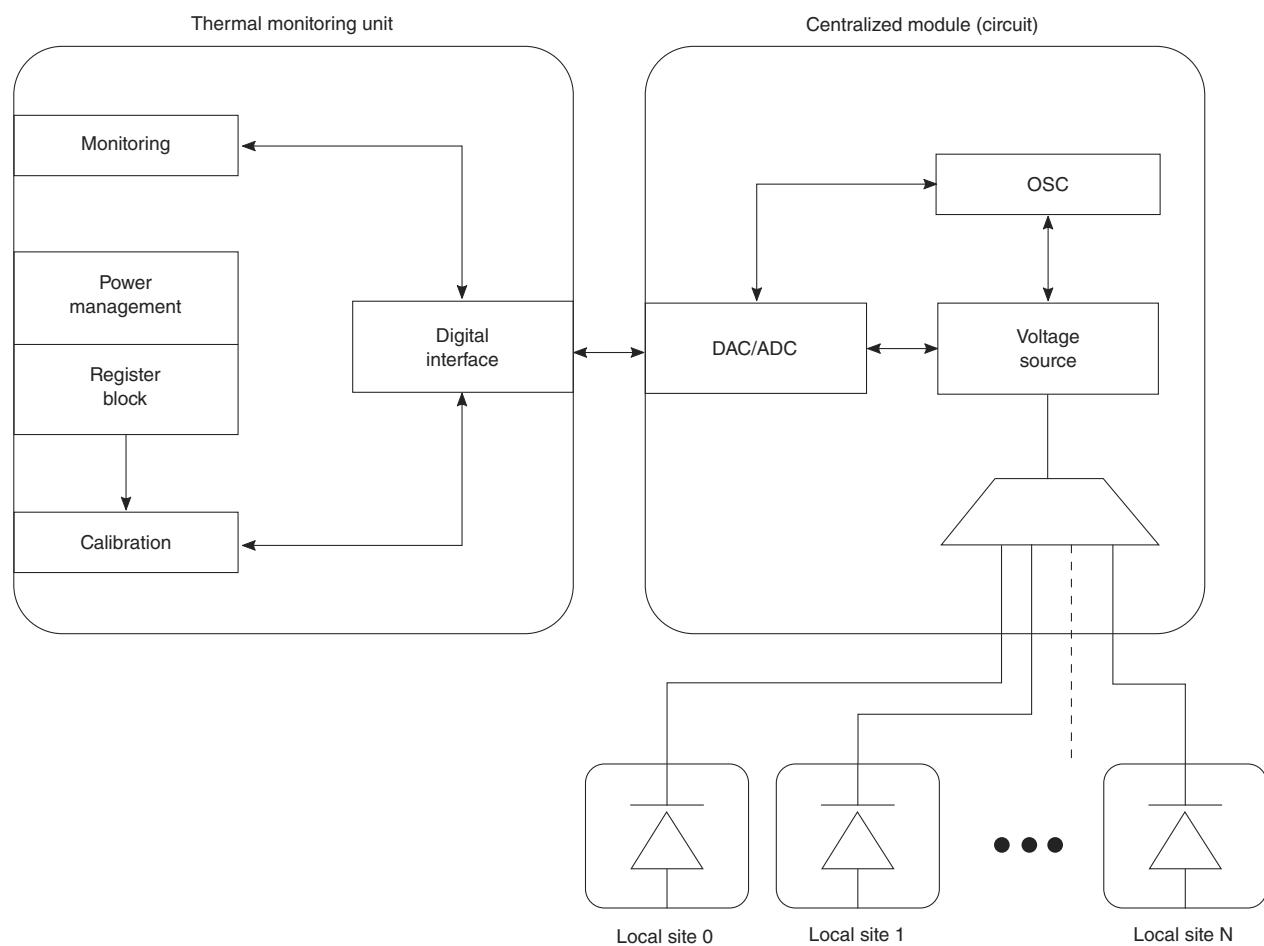
## 35.2 Thermal Monitoring Unit Introduction

The Thermal Monitoring Unit (TMU) monitors and reports the temperature from one or more remote temperature measurement sites located on chip.

### 35.2.1 TMU Overview

The TMU has access to multiple temperature measurement sites strategically located on the chip. It monitors these sites and can signal an alarm if a programmed threshold is ever exceeded. The upper and lower temperature range is continuously captured. A set of reporting registers allow for reading the current temperature at monitored sites.

## Thermal Monitoring Unit Introduction



**Figure 35-1. Thermal Monitoring Unit Block Diagram**

### 35.2.2 Features

The temperature management unit features:

- Temperature measurement range 0-125°C.
- Calibration
  - Calibration table loaded from boot code ROM using pre-boot loader *or* initialization of calibration table through software
- Monitoring
  - Single-, or multi-site monitoring
  - Programmable monitoring interval
  - Out-of-range indication
  - High/low temperature range monitoring
  - Immediate and average temperature monitoring

- Average temperature monitoring programmable low-pass filtering
- Programmable monitoring thresholds for normal and critical alarm
- Reporting
  - Immediate and average temperature reporting for all monitor sites

### 35.2.3 Modes of Operation

The TMU has one mode of operation:

- Monitoring

The mode register monitoring enable bit, TMR[ME], determines if the unit is in active monitoring mode or in power saving mode.

The table below describes bit settings required for each TMU mode of operation.

**Table 35-3. TMU Mode Bit Setting**

Modes with Features	TMR[ME]
Monitoring mode disabled	0
Monitoring mode enabled	1

## 35.3 TMU register descriptions

The table shows the memory map for management of the TMU resources.

### 35.3.1 TMU memory map

TMU base address: 1F0\_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TMU mode register (TMR)	32	RW	0000_0000h
4h	TMU status register (TSR)	32	RO	0000_0000h
8h	TMU monitor temperature measurement interval register (TMTMIR)	32	RW	0000_0000h
20h	TMU interrupt enable register (TIER)	32	RW	0000_0000h
24h	TMU interrupt detect register (TIDR)	32	W1C	0000_0000h

*Table continues on the next page...*

## TMU register descriptions

Offset	Register	Width (In bits)	Access	Reset value
28h	TMU interrupt site capture register (TISCR)	32	RW	0000_0000h
2Ch	TMU interrupt critical site capture register (TICSCR)	32	RW	0000_0000h
40h	TMU monitor high temperature capture register (TMHTCR)	32	RO	0000_0000h
44h	TMU monitor low temperature capture register (TMLTCR)	32	RO	0000_0000h
50h	TMU monitor high temperature immediate threshold register (TMHT ITR)	32	RW	0000_0000h
54h	TMU monitor high temperature average threshold register (TMHT ATR)	32	RW	0000_0000h
58h	TMU monitor high temperature average critical threshold register (TMHTACTR)	32	RW	0000_0000h
80h	TMU temperature configuration register (TTCFGR)	32	RW	0000_0000h
84h	TMU sensor configuration register (TSCFGR)	32	RW	0000_0000h
100h	TMU report immediate temperature site register 0 (TRITSR0)	32	RO	0000_0000h
104h	TMU report average temperature site register 0 (TRATSR0)	32	RO	0000_0000h
110h	TMU report immediate temperature site register 1 (TRITSR1)	32	RO	0000_0000h
114h	TMU report average temperature site register 1 (TRATSR1)	32	RO	0000_0000h
120h	TMU report immediate temperature site register 2 (TRITSR2)	32	RO	0000_0000h
124h	TMU report average temperature site register 2 (TRATSR2)	32	RO	0000_0000h
130h	TMU report immediate temperature site register 3 (TRITSR3)	32	RO	0000_0000h
134h	TMU report average temperature site register 3 (TRATSR3)	32	RO	0000_0000h
140h	TMU report immediate temperature site register 4 (TRITSR4)	32	RO	0000_0000h
144h	TMU report average temperature site register 4 (TRATSR4)	32	RO	0000_0000h
F10h	TMU temperature range 0 control register (TTR0CR)	32	RW	000B_0000h
F14h	TMU temperature range 1 control register (TTR1CR)	32	RW	0009_002Ah
F18h	TMU temperature range 2 control register (TTR2CR)	32	RW	0006_004Ch
F1Ch	TMU temperature range 3 control register (TTR3CR)	32	RW	0007_0062h

## 35.3.2 TMU mode register (TMR)

### 35.3.2.1 Offset

Register	Offset
TMR	0h

### 35.3.2.2 Function

The TMU mode register allows software to control the operation of the thermal monitoring.

### 35.3.2.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ME	Reserved			ALPF		Reserved									
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	MSITE						Reserved									
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.2.4 Fields

Field	Function
0 ME	Monitoring mode enable. 0 No monitoring. 1 Monitoring of sites as defined by field MSITE. Before enabling the TMU for monitoring, the TMU must be configured, see section Initialization Information. Failure to properly initialize the configuration table may result in boundedly undefined behavior.
1-3 —	Reserved
4-5 ALPF	Average low pass filter setting. 00 1.0 01 0.5 10 0.25 11 0.125 The average temperature is calculated as: ALPF x Current_Temp + (1 - ALPF) x Average_Temp. If no previous (average) temperature is valid, current temperature is used. For proper operation, this field should only change when monitoring is disabled.
6-15 —	Reserved

Table continues on the next page...

## TMU register descriptions

Field	Function
16-20 MSITE	Monitoring site select 0 - 4. By setting the select bit for a temperature sensor site, it is enabled and included in all monitoring functions. For proper operation, this field should only change when monitoring is disabled. If no site is selected, site 0 is monitored by default.
21-31 —	Reserved

## 35.3.3 TMU status register (TSR)

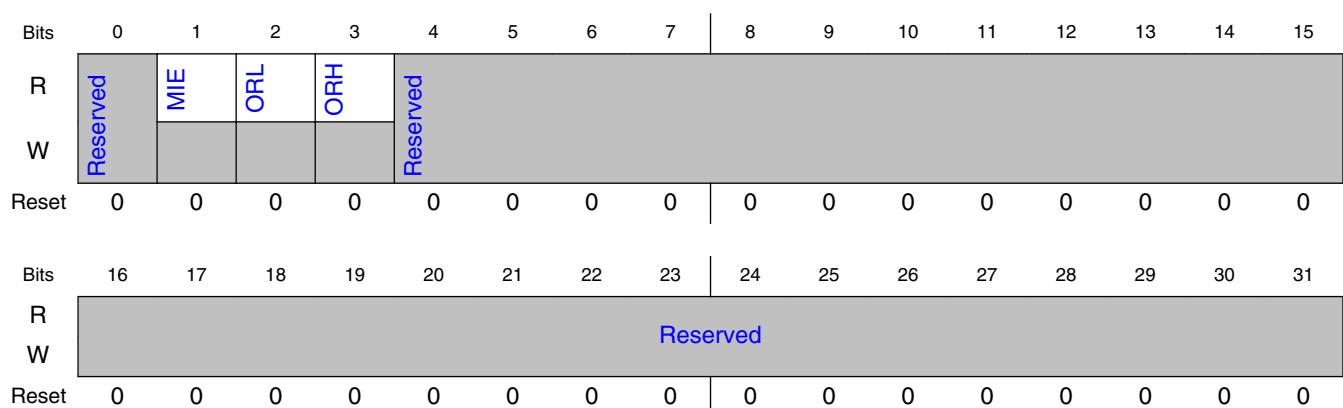
### 35.3.3.1 Offset

Register	Offset
TSR	4h

### 35.3.3.2 Function

The TMU status register reports the monitoring and calibration status during operation.

### 35.3.3.3 Diagram



### 35.3.3.4 Fields

Field	Function
0 —	Reserved
1 MIE	<p>Monitoring interval exceeded.</p> <p>0 Monitoring interval not exceeded.</p> <p>1 Monitoring interval exceeded. The time required to perform measurement of all monitored sites has exceeded the monitoring interval as defined by TMTMIR.</p> <p>This bit will clear automatically when TMU monitoring is (re-)enabled or the monitoring interval register, TMTMIR, is written.</p>
2 ORL	Out-of-range low temperature measurement detected. A temperature sensor detected a temperature reading below the lowest measurable temperature of 0 degrees Celsius. This bit will clear automatically when TMU monitoring is (re-)enabled.
3 ORH	Out-of-range high temperature measurement detected. A temperature sensor detected a temperature reading above the highest measurable temperature of 125 °C. This bit will clear automatically when TMU monitoring is (re-)enabled.
4-31 —	Reserved

## 35.3.4 TMU monitor temperature measurement interval register (TMTMIR)

### 35.3.4.1 Offset

Register	Offset
TMTMIR	8h

### 35.3.4.2 Function

The TMU monitor temperature measurement interval register determines at what frequency temperature sensors are read. All enabled monitored sites are read once in the duration of the time interval. The status bit TSR[MIE] will be set if the temperature measurement takes longer than the set interval. Software should consider increasing the interval or reducing the number of active sites if the interval is exceeded. Disabling the interval allows for continuous monitoring.

**NOTE**

The time it takes for one temperature measurement is dependent on the temperatures measured and mode settings, thus there is no fixed time advertised for a single temperature measurements.

The temperature measurement interval are determined using the formula,  $[2^{(22+TMI)} / \text{Platform frequency in HZ}]$ . For supported platform frequencies options, refer to data sheet. For example the following table lists temperature measurement intervals as calculated by formula.

**Table 35-4. Temperature measurement interval**

TMI Bit Field Setting	Platform Clock Frequency		
	256MHz	333MHz	400MHz
0000	0.016 s	0.013 s	0.01 s
0001	0.03 s	0.025 s	0.02 s
0010	0.07 s	0.05 s	0.04 s
0011	0.13 s	0.1 s	0.06 s
0100	0.26 s	0.2 s	0.17 s
0101	0.5 s	0.4 s	0.34 s
0110	1 s	0.8 s	0.67 s
0111	2.1 s	1.6 s	1.34 s
1000	4.2 s	3.2 s	2.7 s
1001	8.4 s	6.4 s	5.4 s
1010	16.7 s	12.9 s	10.7 s
1011	33.6 s	25.8 s	21.5 s
1100	67.1 s	51.6 s	42.9 s
1101	134.2 s	103.2 s	85.9 s
1110	268.4 s	206.4 s	171.8 s
1111	Disabled		

**35.3.4.3 Diagram**

Bits	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15
R	Reserved																
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	
Bits	16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
R	Reserved												TMI				
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	

### 35.3.4.4 Fields

Field	Function
0-27 —	Reserved
28-31 TMI	Temperature monitoring interval in seconds. For proper operation, this field should only change when monitoring is disabled, TMR[ME]=0. For lower platform speeds, the time increases proportionally, while the opposite is true for higher platform speeds.

### 35.3.5 TMU interrupt enable register (TIER)

#### 35.3.5.1 Offset

Register	Offset
TIER	20h

#### 35.3.5.2 Function

The TMU interrupt enable register determines if a detected status condition should cause a system interrupt. A system interrupt occurs if a bit in this register is set and the corresponding bit in the interrupt detect register is also set. To clear the interrupt, write a 1 to the interrupt detect register.

### 35.3.5.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R																
W	ITTEIE E	ATTEIE	ATCTEIE	Reserved												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R									Reserved							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.5.4 Fields

Field	Function
0 ITTEIE	Immediate temperature threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ITTE] is set.
1 ATTEIE	Average temperature threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ATTE] is set.
2 ATCTEIE	Average temperature critical threshold exceeded interrupt enable. 0 Disabled. 1 Interrupt enabled. Generate an interrupt if TIDR[ATCTE] is set.
3-31 —	Reserved

## 35.3.6 TMU interrupt detect register (TIDR)

### 35.3.6.1 Offset

Register	Offset
TIDR	24h

### 35.3.6.2 Function

The TMU interrupt detect register indicates if an status condition was detected that could generate an interrupt. Write 1 to clear the detected condition and the interrupt, if enabled.

### 35.3.6.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ITTE	ATTE	ATCTE	Reserved												
W	W1C	W1C	W1C	Reserved												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.6.4 Fields

Field	Function
0 ITTE	Immediate temperature threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Immediate temperature threshold, as defined by TMHTITR, has been exceeded by one or more monitored sites. This includes an out-of-range measured temperature above 125 °C. The sites which has exceeded the threshold are captured in TISCR[ISITE].
1 ATTE	Average temperature threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Average temperature threshold, as defined by TMHTATR, has been exceeded by one or more monitored sites. The sites which has exceeded the threshold are captured in TISCR[ASITE].
2 ATCTE	Average temperature critical threshold exceeded. Write 1 to clear. 0 No threshold exceeded. 1 Average temperature critical threshold, as defined by TMHTACTR, has been exceeded by one or more monitored sites. The sites which has exceeded the threshold are captured in TICSCR[CASITE].
3-31 —	Reserved

## 35.3.7 TMU interrupt site capture register (TISCR)

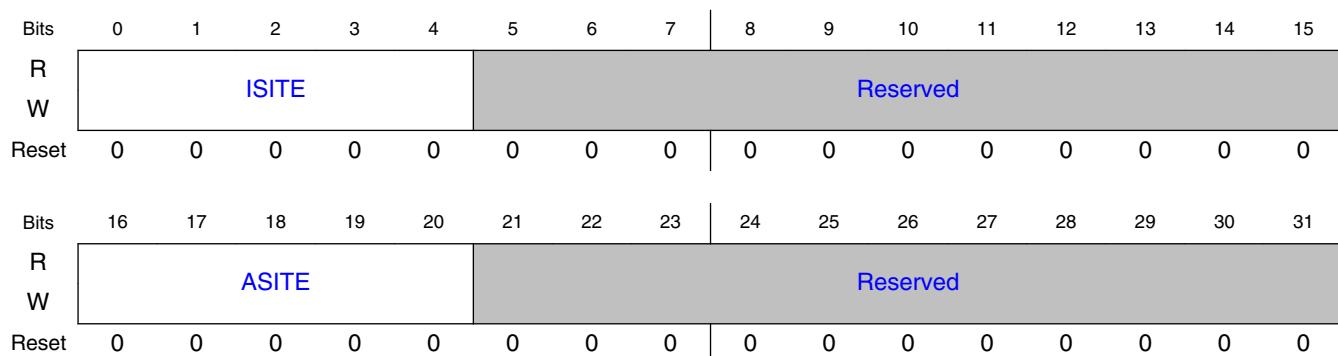
### 35.3.7.1 Offset

Register	Offset
TISCR	28h

### 35.3.7.2 Function

The TMU interrupt site capture register holds information about the temperature sensor site associated with a detected interrupt event.

### 35.3.7.3 Diagram



### 35.3.7.4 Fields

Field	Function
0-4 ISITE	Temperature sensor site associated with the setting of TIDR[ITTE]. This field has the same bit representation as TMR[MSITE]. Software should clear this field after handling the detected interrupt events ITTE.
5-15 —	Reserved
16-20 ASITE	Temperature sensor site associated with the setting of TIDR[ATTE] . This field has the same bit representation as TMR[MSITE]. Software should clear this field after handling the detected interrupt event ATTE.

Table continues on the next page...

Field	Function
21-31	Reserved
—	

### 35.3.8 TMU interrupt critical site capture register (TICSCR)

#### 35.3.8.1 Offset

Register	Offset
TICSCR	2Ch

#### 35.3.8.2 Function

The TMU interrupt critical site capture register holds information about the temperature sensor site associated with a detected critical interrupt event.

#### 35.3.8.3 Diagram



#### 35.3.8.4 Fields

Field	Function
0-15	Reserved
—	

Table continues on the next page...

## TMU register descriptions

Field	Function
16-20 CASITE	Temperature sensor site associated with the setting of TIDR[ATCTE]. This field has the same bit representation as TMR[MSITE]. Software should clear this field after handling the detected critical interrupt event ATCTE.
21-31 —	Reserved

## 35.3.9 TMU monitor high temperature capture register (TMHTCR)

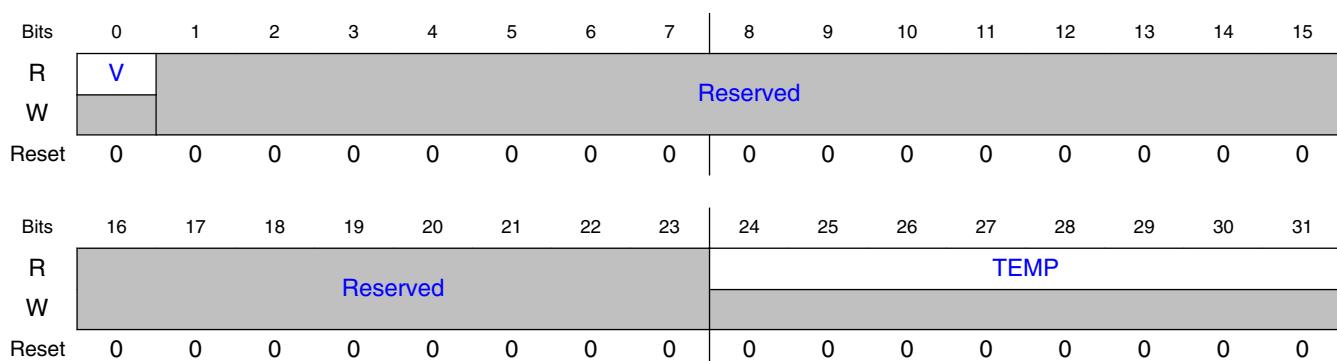
### 35.3.9.1 Offset

Register	Offset
TMHTCR	40h

### 35.3.9.2 Function

This TMU monitor register captures and record the highest temperature reached for any one enabled monitored site within temperature sensor range.

### 35.3.9.3 Diagram



### 35.3.9.4 Fields

Field	Function
0	Valid reading.
V	0 Temperature reading is not valid due to no measured temperature within the sensor range of 0-125 °C for an enabled monitored site. 1 Temperature reading is valid. (Re-)enabling the TMU will automatically clear this bit and start a new search.
1-23	Reserved
—	
24-31	Highest temperature recorded in degrees Celcius by any enabled monitored site. Valid when V=1.
TEMP	0-125 °C Sensor range 126-255 °C Reserved

### 35.3.10 TMU monitor low temperature capture register (TMLTCR)

#### 35.3.10.1 Offset

Register	Offset
TMLTCR	44h

#### 35.3.10.2 Function

This TMU monitor register captures and record the lowest temperature reached for any one enabled monitored site within temperature sensor range.

### 35.3.10.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	V								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved					TEMP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.10.4 Fields

Field	Function
0	Valid reading.
V	0 Temperature reading is not valid due to no measured temperature within the sensor range of 0-125 °C for an enabled monitored site. 1 Temperature reading is valid. (Re-)enabling the TMU will automatically clear this bit and start a new search.
1-23 —	Reserved
24-31	Lowest temperature recorded in degrees Celcius by any enabled monitored site. Valid when V=1.
TEMP	0-125 °C Sensor range 126-255 °C Reserved

### 35.3.11 TMU monitor high temperature immediate threshold register (TMHTITR)

#### 35.3.11.1 Offset

Register	Offset
TMHTITR	50h

### 35.3.11.2 Function

This TMU monitor register determines the high current temperature threshold for generating the TIDR[ITTE] event.

### 35.3.11.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EN								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved										TEMP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.11.4 Fields

Field	Function
0 EN	Enable threshold. 0 Disabled. 1 Threshold enabled.
1-23 —	Reserved
24-31 TEMP	High temperature immediate threshold value. Determines the current upper temperature threshold, for any enabled monitored site, that if exceeded will cause TIDR[ITTE] to be set when EN=1. 0-125 °C Sensor range 126-255 °C Reserved

### 35.3.12 TMU monitor high temperature average threshold register (TMHTATR)

### 35.3.12.1 Offset

Register	Offset
TMHTATR	54h

### 35.3.12.2 Function

This TMU monitor register determines the high average temperature threshold for generating the TIDR[ATTE] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating average temperature.

### 35.3.12.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	EN								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved										TEMP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.12.4 Fields

Field	Function
0	Enable threshold.
EN	0 Disabled. 1 Threshold enabled.
1-23	Reserved
—	
24-31	High temperature average threshold value. Determines the average upper temperature threshold, for any enabled monitor site, that if exceeded will cause TIDR[ATTE] to be set when EN=1.
TEMP	0-125 °C Sensor range 126-255 °C Reserved

### 35.3.13 TMU monitor high temperature average critical threshold register (TMHTACTR)

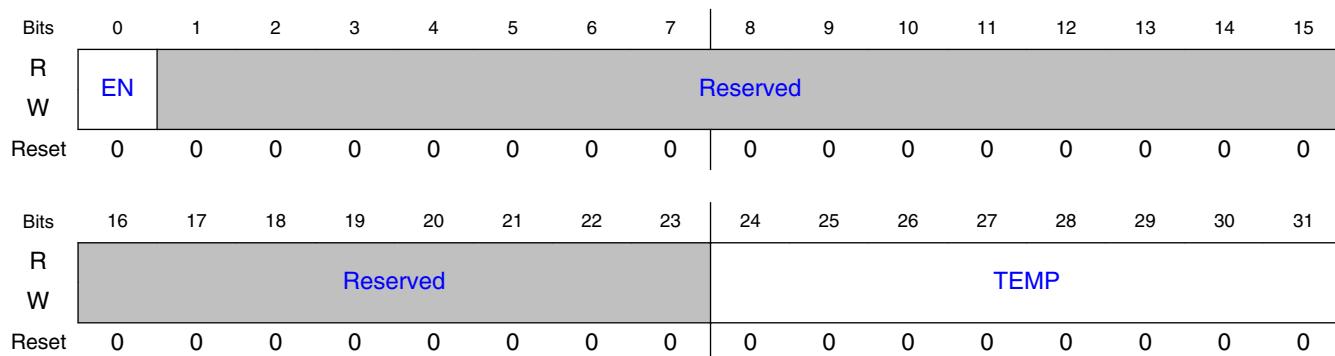
#### 35.3.13.1 Offset

Register	Offset
TMHTACTR	58h

#### 35.3.13.2 Function

This TMU monitor register determines the high average critical temperature threshold for generating the TIDR[ATCTE] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating average temperature.

#### 35.3.13.3 Diagram



#### 35.3.13.4 Fields

Field	Function
0	Enable threshold.
EN	0 Disabled. 1 Threshold enabled.
1-23	Reserved

Table continues on the next page...

## TMU register descriptions

Field	Function
—	
24-31 TEMP	High temperature average critical threshold value. Determines the average upper critical temperature threshold, for any enabled monitored site, that if exceeded will cause TIDR[ATCTE] to be set when EN=1. 0-125 °C Sensor range 126-255 °C Reserved

### 35.3.14 TMU temperature configuration register (TTCFGR)

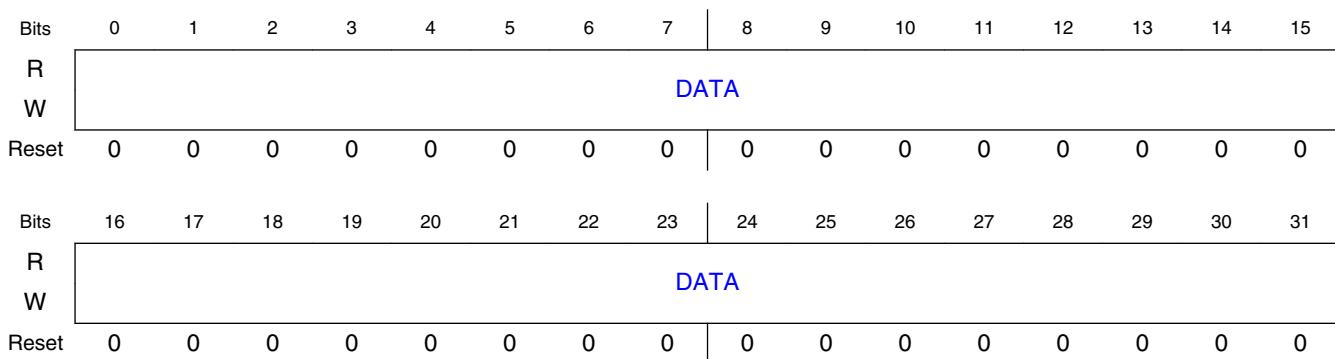
#### 35.3.14.1 Offset

Register	Offset
TTCFGR	80h

#### 35.3.14.2 Function

The TMU temperature configuration register, in conjunction with the sensor configuration register, is used to initialize the internal sensor translation table used during monitoring. This register pair defines indirect access to the table. See the section Initialization Information.

#### 35.3.14.3 Diagram



### 35.3.14.4 Fields

Field	Function
0-31	Sensor data.
DATA	

## 35.3.15 TMU sensor configuration register (TSCFGR)

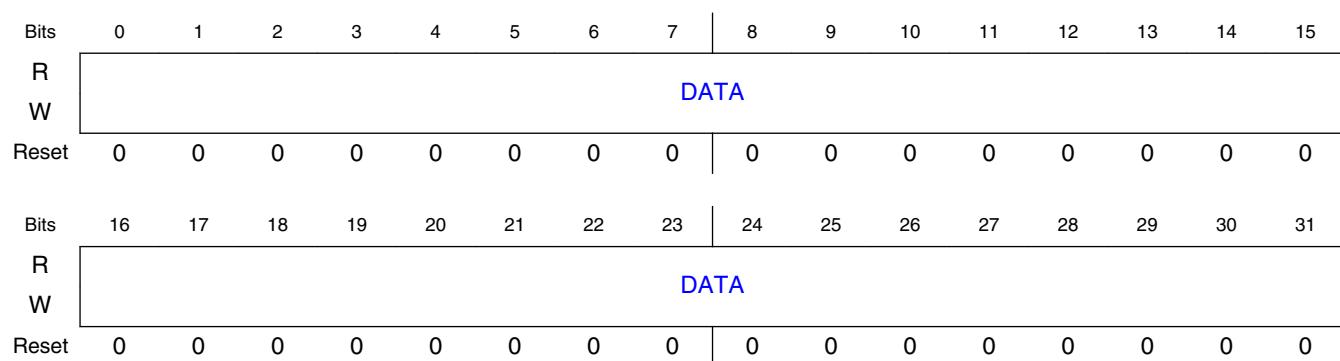
### 35.3.15.1 Offset

Register	Offset
TSCFGR	84h

### 35.3.15.2 Function

The TMU sensor configuration register, in conjunction with the temperature configuration register, is used to initialize the internal sensor translation table used during monitoring. This register pair defines indirect access to the table. Reading this register will return the data from the translation table as defined by TTCFGR. See the section Initialization Information.

### 35.3.15.3 Diagram



### 35.3.15.4 Fields

Field	Function
0-31 DATA	Sensor data.

## 35.3.16 TMU report immediate temperature site register a (TRITSR0 - TRITSR4)

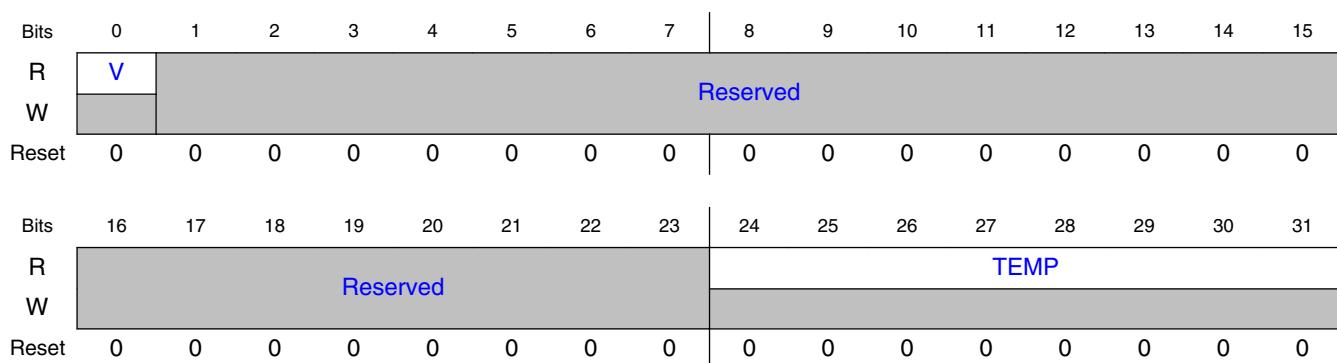
### 35.3.16.1 Offset

Register	Offset
TRITSR0	100h
TRITSR1	110h
TRITSR2	120h
TRITSR3	130h
TRITSR4	140h

### 35.3.16.2 Function

This TMU report register returns the last measured temperature at site. The site must be part of the list of enabled monitored sites as defined by TMR[MSITE].

### 35.3.16.3 Diagram



### 35.3.16.4 Fields

Field	Function
0	Valid measured temperature.
V	0 Not valid. Temperature out of sensor range or first measurement still pending. 1 Valid.
1-23	Reserved
—	
24-31	Last temperature reading at site when V=1.
TEMP	

### 35.3.17 TMU report average temperature site register a (TRAT SR0 - TRATSR4)

#### 35.3.17.1 Offset

Register	Offset
TRATSR0	104h
TRATSR1	114h
TRATSR2	124h
TRATSR3	134h
TRATSR4	144h

#### 35.3.17.2 Function

This TMU report register returns the average measured temperature at site. The site must be part of the list of enabled monitored sites as defined by TMR[MSITE].

### 35.3.17.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	V								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R				Reserved					TEMP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 35.3.17.4 Fields

Field	Function
0	Valid measured temperature.
V	0 Not valid. Temperature out of sensor range or first measurement still pending. 1 Valid.
1-23	Reserved
—	
24-31	Average temperature reading at site when V=1.
TEMP	

## 35.3.18 TMU temperature range 0 control register (TTR0CR)

### 35.3.18.1 Offset

Register	Offset
TTR0CR	F10h

### 35.3.18.2 Function

The TMU temperature range control registers allow for defining the four temperature ranges that each sensor covers. A starting temperature range and number of calibration points, up to 16, is given. The temperature ranges may overlap to remove possible sample error, when searching for an accurate sensor reading. The default temperature calibration points are shown below.

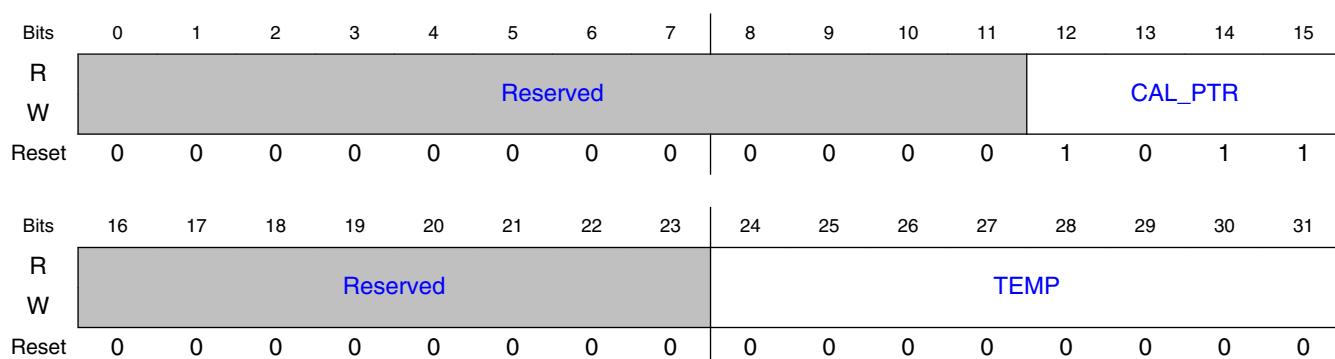
#### NOTE

The circuit is designed and calibrated for measuring temperatures in the range of 0C to 125C. It is not advisable to go beyond this range when setting up the calibration points without prior circuit knowledge. The margin of error cannot be guaranteed outside these limits.

**Table 35-5. Default Temperature Configuration Points**

Configuration Range	Starting Temperature (Celsius)	Temperature Configuration Points (Celsius)	Number of Points
0	0	0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	12
1	42	42, 46, 50, 54, 58, 62, 66, 70, 74, 78	10
2	76	76, 80, 84, 88, 92, 96, 100	7
3	98	98, 102, 106, 110, 114, 118, 122, 126	8

### 35.3.18.3 Diagram



### 35.3.18.4 Fields

Field	Function
0-11	Reserved

*Table continues on the next page...*

## TMU register descriptions

Field	Function
—	
12-15 CAL_PTR	Temperature configuration points. 0000 Reserved 0001 2 points 0010 3 points ... 1111 16 points
16-23	Reserved
—	
24-31 TEMP	Starting temperature in Celsius for range.

## 35.3.19 TMU temperature range 1 control register (TTR1CR)

### 35.3.19.1 Offset

Register	Offset
TTR1CR	F14h

### 35.3.19.2 Function

The TMU temperature range control registers allow for defining the four temperature ranges that each sensor covers. A starting temperature range and number of calibration points, up to 16, is given. The temperature ranges may overlap to remove possible sample error, when searching for an accurate sensor reading. The default temperature calibration points are shown below.

#### NOTE

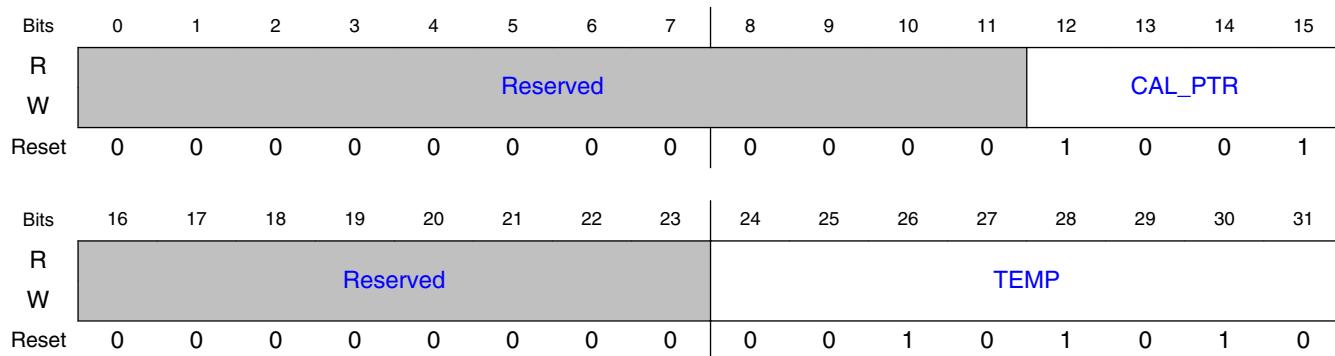
The circuit is designed and calibrated for measuring temperatures in the range of 0C to 125C. It is not advisable to go beyond this range when setting up the calibration points

without prior circuit knowledge. The margin of error cannot be guaranteed outside these limits.

**Table 35-6. Default Temperature Configuration Points**

Configuration Range	Starting Temperature (Celsius)	Temperature Configuration Points (Celsius)	Number of Points
0	0	0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	12
1	42	42, 46, 50, 54, 58, 62, 66, 70, 74, 78	10
2	76	76, 80, 84, 88, 92, 96, 100	7
3	98	98, 102, 106, 110, 114, 118, 122, 126	8

### 35.3.19.3 Diagram



### 35.3.19.4 Fields

Field	Function
0-11 —	Reserved
12-15 CAL_PTR	Temperature configuration points. 0000 Reserved 0001 2 points 0010 3 points ... 1111 16 points
16-23 —	Reserved
24-31 TEMP	Starting temperature in Celsius for range.

## 35.3.20 TMU temperature range 2 control register (TTR2CR)

### 35.3.20.1 Offset

Register	Offset
TTR2CR	F18h

### 35.3.20.2 Function

The TMU temperature range control registers allow for defining the four temperature ranges that each sensor covers. A starting temperature range and number of calibration points, up to 16, is given. The temperature ranges may overlap to remove possible sample error, when searching for an accurate sensor reading. The default temperature calibration points are shown below.

#### NOTE

The circuit is designed and calibrated for measuring temperatures in the range of 0C to 125C. It is not advisable to go beyond this range when setting up the calibration points without prior circuit knowledge. The margin of error cannot be guaranteed outside these limits.

**Table 35-7. Default Temperature Configuration Points**

Configuration Range	Starting Temperature (Celsius)	Temperature Configuration Points (Celsius)	Number of Points
0	0	0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	12
1	42	42, 46, 50, 54, 58, 62, 66, 70, 74, 78	10
2	76	76, 80, 84, 88, 92, 96, 100	7
3	98	98, 102, 106, 110, 114, 118, 122, 126	8

### 35.3.20.3 Diagram

Bits	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved								CAL_PTR							
W	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
Bits	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved								TEMP							
W	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

### 35.3.20.4 Fields

Field	Function
0-11	Reserved
—	
12-15 CAL_PTR	Temperature configuration points. 0000 Reserved 0001 2 points 0010 3 points ... 1111 16 points
16-23	Reserved
—	
24-31 TEMP	Starting temperature in Celsius for range.

## 35.3.21 TMU temperature range 3 control register (TTR3CR)

### 35.3.21.1 Offset

Register	Offset
TTR3CR	F1Ch

### 35.3.21.2 Function

The TMU temperature range control registers allow for defining the four temperature ranges that each sensor covers. A starting temperature range and number of calibration points, up to 16, is given. The temperature ranges may overlap to remove possible sample error, when searching for an accurate sensor reading. The default temperature calibration points are shown below.

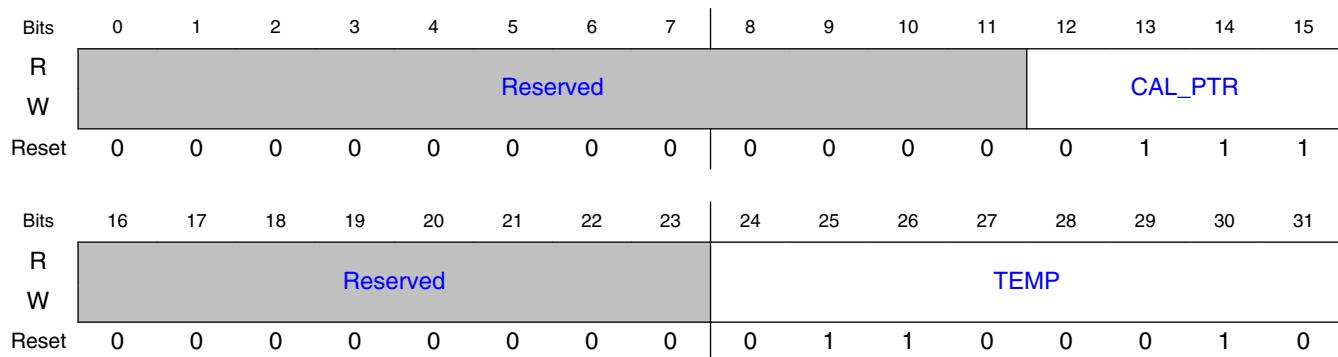
#### NOTE

The circuit is designed and calibrated for measuring temperatures in the range of 0C to 125C. It is not advisable to go beyond this range when setting up the calibration points without prior circuit knowledge. The margin of error cannot be guaranteed outside these limits.

**Table 35-8. Default Temperature Configuration Points**

Configuration Range	Starting Temperature (Celsius)	Temperature Configuration Points (Celsius)	Number of Points
0	0	0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44	12
1	42	42, 46, 50, 54, 58, 62, 66, 70, 74, 78	10
2	76	76, 80, 84, 88, 92, 96, 100	7
3	98	98, 102, 106, 110, 114, 118, 122, 126	8

### 35.3.21.3 Diagram



### 35.3.21.4 Fields

Field	Function
0-11 —	Reserved
12-15 CAL_PTR	Temperature configuration points. 0000 Reserved 0001 2 points 0010 3 points ... 1111 16 points
16-23 —	Reserved
24-31 TEMP	Starting temperature in Celsius for range.

## 35.4 Functional Description

The following sections describe the functionality of the TMU in details.

### 35.4.1 Monitoring

Monitoring is the process of reading enabled temperature sensor sites on-chip at regular intervals and taking appropriate actions, such as alarming the user when the temperature exceeds a programmed temperature threshold.

During monitoring, all enabled temperature sensor sites are periodically measured for temperature starting with site 0 and ending with site 4. The interval at which the sensors are read is set in TMTMIR and should reflect the maximum interval time required to accurately capture temperature changes. If the measurement interval has not expired after the last active site has been read, the sensor logic enters low-power mode. If the interval has expired when the last active site is read, the measurement interval exceeded bit, TSR[MIE], is set and the next active site is read immediately. If the interval has been exceeded, user may opt to reduce number of sites monitored or increase the interval, if possible.

For each site the current and average temperature is logged. The average temperature is calculated based on the low-pass filter function in the mode register. If any of the set temperature threshold registers are exceeded, the corresponding interrupt detect bit is set in TIDR. Interrupts are enabled through the interrupt enable register, TIER.

Process for enabling monitoring mode:

1. Clear the interrupt detect register, TIDR.
2. Clear interrupt site capture register (TISCR) and interrupt critical site capture register (TICSCR).
3. Enable interrupt handling by setting the appropriate bits in TIER.
4. Set the temperature threshold registers TMHTITR, TMHTATR and TMHTACTR.
5. Set the monitoring interval register, TMTMIR.
6. Enable monitor mode by setting TMR[ME]=1. Sites to monitor are controlled by setting TMR[MSITE]. There should be at least one active site enabled. Set other mode control bits as needed.
7. If the monitoring interval is too short as indicated by TSR[MIE], the interval may need to be increased or number of sites reduced.

### **35.4.2 Reporting**

The TMU can directly report the current and average temperature for a particular temperature sensor site during monitoring mode by reading one of the report registers per site. The report uses the last measurement done by the monitoring process and requires a site to be actively monitored for accurate reading. Reading a site which has last measured an invalid temperature outside the sensor range of 0-125 degrees Celsius, will have the valid bit cleared.

If monitoring is disabled, the last temperature measurement remains for the site(s) previously monitored and can still be read using the report registers knowing that the temperature reported is no longer accurate. This method can be used to capture the temperature at multiple sites in time, but does not allow for continuous monitoring.

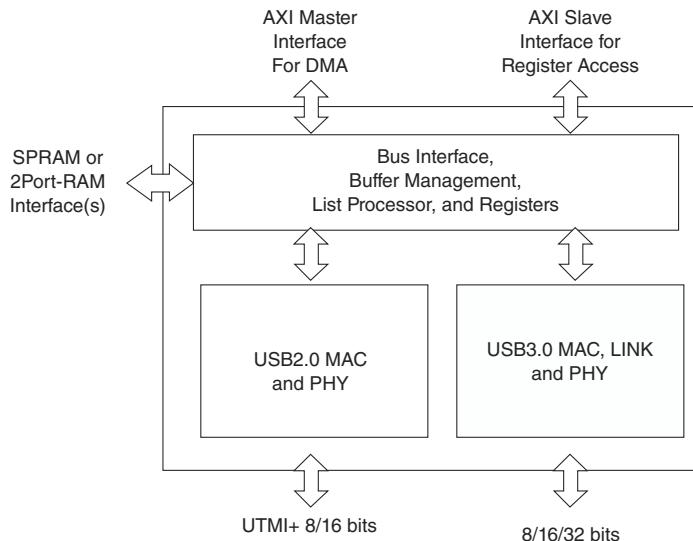
# Chapter 36

## Universal Serial Bus Interface 3.0

### 36.1 Overview

The USB module is a USB 3.0-compliant serial interface engine for implementing a USB interface. This module may be connected to an external port. Collectively the module and external port are called the USB 3.0 interface. USB 3.0 supports super-speed (SS), high-speed (HS), full-speed (FS), and low-speed (LS) operations.

- The upper layer is common for USB 2.0 and USB 3.0 operation. This has the bus interface, buffer management block, list processor for scheduling, and control and status register (CSR) functions
- USB 2.0 PHY and MAC layers
- USB 3.0 PHY, LINK, and MAC layers



**Figure 36-1. USB interface block diagram**

### 36.1.1 Features

The USB 3.0 module includes the following features:

- Complies with USB specification rev 3.0 (xHCI compatible)
- Supports operation as a standalone USB host controller
- USB dual-role operation and can be configured as host or device
- Super-speed (5 Gbit/s), high-speed (480 Mbit/s), full-speed (12 Mbit/s), and low-speed (1.5 Mbit/s) operations.
- Supports operation as a standalone single port USB
- Supports four programmable, bidirectional USB endpoints
- OTG (on-the-go) 2.0 compliant, which includes both device and host capability. Super-speed operation is not supported when OTG is enabled.
- Supports system memory interface with 40-bit addressing capability

### 36.1.2 Modes of Operation

The USB 3.0 module operates in following modes.

- Host Mode: SS/HS/FS/LS
- Device Mode: SS/HS/FS
- OTG: HS/FS/LS

### 36.1.3 External Signals

This section contains detailed descriptions of all the USB 3.0 controller signals. Many of the signals for the PHY interfaces are muxed onto the same pins in order to reduce pin count. The following table shows the USB signals, indicating which interface supports each signal.

**Table 36-1. USB 3.0 External Signals**

Signal	I/O	Description
USBn_D_P	IO	USB PHY Data Plus
USBn_D_M	IO	USB PHY Data Minus
USBn_VBUS	I	USB PHY VBUS
USBn_ID	IO	USB PHY ID Detect
USBn_TX_P	O	USB PHY 3.0 Transmit Data (positive)
USBn_TX_M	O	USB PHY 3.0 Transmit Data (negative)

*Table continues on the next page...*

**Table 36-1. USB 3.0 External Signals (continued)**

Signal	I/O	Description
USBn_RX_P	I	USB PHY 3.0 Receive Data (positive)
USBn_RX_M	I	USB PHY 3.0 Receive Data (negative)
USBn_RESREF	IO	USB PHY Impedance Calibration
USBn_DRVVBUS	O	VBus power enable.
USBn_PWRFAULT	I	Indicates that a Vbus fault has occurred.

## 36.2 USB Memory Map/Register Definition

USB3.0 registers are 32 bits wide, and the addresses are 32-bit block aligned. To avoid hardware or software incompatibility, the driver must access the registers as 32-bit units. The driver must not access these registers as 8-bit or 16-bit. CSRs are classified as follows:

**Table 36-2. Registers**

Control and Status Registers	Reference
Global registers	<a href="#">Capability Registers Length and HC Interface Version Number (USB_CAPLENGTH)</a> to <a href="#">Global Frame Length Adjustment Register (USB_GFLADJ)</a>
Device register	<a href="#">Device Configuration Register (USB_DCFG)</a> to <a href="#">Device Physical Endpoint-n Command Parameter 0 Register (USB_DEPCMDPAR0n)</a>
OTG register	<a href="#">OTG Configuration Register (USB_OCFG)</a> to <a href="#">ADP Event Enable Register (USB_ADPEVTEN)</a>
xHCI host register	See xHCI Spec for details. Refer <a href="#">Table 36-3</a> for list of registers.

The following table lists the supported xHCI registers. For register definitions, refer to the xHCI specification.

**Table 36-3. Host registers**

OFFSET	Register	OFFSET	Register	
Register Set's Base= 0x0000		Register Set's Base= DBOFF		
0x0000	CAPLENGTH	0x0000	DB	
0x0004	HCSPARAMS1	Register Set's Base= xECP * 4		
0x0008	HCSPARAMS2	0x0000	USBLEGSUP	
0x000C	HCSPARAMS3	0x0004	USBLEGCTLSTS	
0x0010	HCCPARAMS	Addr1		
0x0014	DBOFF	0x0000	SUPTPRT2_DW0	

*Table continues on the next page...*

**Table 36-3. Host registers (continued)**

OFFSET	Register	OFFSET	Register
0x0018	RTSOFF	0x0004	SUPTPRT2_DW1
0x001C	Reserved	0x0008	SUPTPRT2_DW2
0x0020	USBCMD	0x000C	SUPTPRT2_DW3
0x0024	USBSTS	Addr2	
0x0028	PAGESIZE	0x0000	SUPTPRT3_DW0
0x002C-0x0033	Reserved	0x0004	SUPTPRT3_DW1
0x0034	DNCTRL	0x0008	SUPTPRT3_DW2
0x0038	CRCR	0x000C	SUPTPRT3_DW3
0x003C	CRCR	Addr2 + 10h	
0x0040-0x004F	Reserved	0x0000	DCID
0x0050	DCBAAP	0x0004	DCDB
0x0054	DCBAAP	0x0008	DCERSTSZ
0x0058	CONFIG	0x000C	Reserved
0x005C-0x041F	Reserved	0x0010	DCERSTBA
0x0420	PORTSC PORTPMSC_SS1	0x0014	DCERSTBA
0x0424	PORTPMSC_201	0x0018	DCERDP
0x0428	PORTLI	0x001C	DCERDP
0x042C	PORTHLPMC_SS2	0x0020	DCCTRL
0x042C	PORHLPMC_202	0x0024	DCST
Register Set's Base= RTSOFF		0x0028	DCPORTSC
0x0000	MFINDEX	0x002C	Reserved
0x0004	Reserved	0x0030	DCCP
Register Set's Base= RTSOFF + 20h		0x0034	DCCP
0x0000	IMAN	0x0038	DCDDI1
0x0004	IMOD	0x003C	DCDDI2
0x0008	ERSTSZ		
0x000C	Reserved		
0x0010	ERSTBA		
0x0014	ERSTBA		
0x0018	ERDP		
0x001C	ERDP		

**USB memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2F0_0000	Capability Registers Length and HC Interface Version Number (USB1_CAPLENGTH)	32	R	<a href="#">See section</a>	<a href="#">36.2.1/2268</a>

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2F0_0004	Host Controller Structural Parameters 1 (USB1_HCSPARAMS1)	32	R	<a href="#">See section</a>	36.2.2/2269
2F0_0008	Host Controller Structural Parameters 2 (USB1_HCSPARAMS2)	32	R	<a href="#">See section</a>	36.2.3/2269
2F0_000C	Host Controller Structural Parameters 3 (USB1_HCSPARAMS3)	32	R	<a href="#">See section</a>	36.2.4/2270
2F0_0010	Host Controller Capability Parameters 1 (USB1_HCCPARAMS1)	32	R	<a href="#">See section</a>	36.2.5/2271
2F0_0014	Doorbell Offset (USB1_DBOFF)	32	R	<a href="#">See section</a>	36.2.6/2272
2F0_0018	Runtime Register Space Offset (USB1_RTSOFF)	32	R	<a href="#">See section</a>	36.2.7/2273
2F0_001C	Host Controller Capability Parameters 2 (USB1_HCCPARAMS2)	32	R	<a href="#">See section</a>	36.2.8/2273
2F0_C100	Global SoC Bus Configuration Register 0 (USB1_GSBUSCFG0)	32	R/W	<a href="#">See section</a>	36.2.9/2274
2F0_C104	Global SoC Bus Configuration Register 1 (USB1_GSBUSCFG1)	32	R/W	<a href="#">See section</a>	36.2.10/2279
2F0_C108	Global Tx Threshold Control Register (USB1_GTXTHRCFG)	32	R/W	0000_0000h	36.2.11/2280
2F0_C10C	Global Rx Threshold Control Register (USB1_GRXTHRCFG)	32	R/W	0000_0000h	36.2.12/2281
2F0_C110	Global Core Control Register (USB1_GCTL)	32	R/W	<a href="#">See section</a>	36.2.13/2283
2F0_C118	Global Status Register (USB1_GSTS)	32	R/W	<a href="#">See section</a>	36.2.14/2288
2F0_C11C	Global User Control Register 1 (USB1_GUCTL1)	32	R/W	<a href="#">See section</a>	36.2.15/2290
2F0_C128	Global User ID Register (USB1_GUID)	32	R	<a href="#">See section</a>	36.2.16/2294
2F0_C12C	Global User Control Register (USB1_GUCTL)	32	R/W	<a href="#">See section</a>	36.2.17/2295
2F0_C130	Global SoC Bus Error Address Register low (USB1_GBUSERRADDRLO)	32	R	0000_0000h	36.2.18/2298
2F0_C134	Global SoC Bus Error Address Register high (USB1_GBUSERRADDRHI)	32	R	0000_0000h	36.2.19/2299
2F0_C138	Global SS Port to Bus Instance Mapping Register - Low (USB1_GPRTBIMAPLO)	32	R/W	0000_0000h	36.2.20/2299
2F0_C13C	Global SS Port to Bus Instance Mapping Register - High (USB1_GPRTBIMAPHI)	32	R	0000_0000h	36.2.21/2300
2F0_C140	Global Hardware Parameters Register 0 (USB1_GHWPARAMS0)	32	R	<a href="#">See section</a>	36.2.22/2300
2F0_C144	Global Hardware Parameters Register 1 (USB1_GHWPARAMS1)	32	R	<a href="#">See section</a>	36.2.23/2302
2F0_C148	Global Hardware Parameters Register 2 (USB1_GHWPARAMS2)	32	R	<a href="#">See section</a>	36.2.24/2305

Table continues on the next page...

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2F0_C14C	Global Hardware Parameters Register 3 (USB1_GHWPARAMS3)	32	R	<a href="#">See section</a>	36.2.25/ 2306
2F0_C150	Global Hardware Parameters Register 4 (USB1_GHWPARAMS4)	32	R	<a href="#">See section</a>	36.2.26/ 2309
2F0_C154	Global Hardware Parameters Register 5 (USB1_GHWPARAMS5)	32	R	<a href="#">See section</a>	36.2.27/ 2311
2F0_C158	Global Hardware Parameters Register 6 (USB1_GHWPARAMS6)	32	R	<a href="#">See section</a>	36.2.28/ 2312
2F0_C15C	Global Hardware Parameters Register 7 (USB1_GHWPARAMS7)	32	R	<a href="#">See section</a>	36.2.29/ 2314
2F0_C180	Global High-Speed Port to Bus Instance Mapping Register - Low (USB1_GPRTBIMAP_HSLO)	32	R/W	0000_0000h	36.2.30/ 2315
2F0_C184	Global High-Speed Port to Bus Instance Mapping Register - High (USB1_GPRTBIMAP_HSHI)	32	R	0000_0000h	36.2.31/ 2315
2F0_C200	Global USB2 PHY Configuration Register (USB1_GUSB2PHYCFG)	32	R/W	<a href="#">See section</a>	36.2.32/ 2316
2F0_C2C0	Global USB 3.0 PIPE Control Register (USB1_GUSB3PIPECTL)	32	R/W	<a href="#">See section</a>	36.2.33/ 2319
2F0_C300	Global Transmit FIFO Size Register (USB1_GTXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
2F0_C304	Global Transmit FIFO Size Register (USB1_GTXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
2F0_C308	Global Transmit FIFO Size Register (USB1_GTXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
2F0_C30C	Global Transmit FIFO Size Register (USB1_GTXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
2F0_C380	Global Receive FIFO Size Register (USB1_GRXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	36.2.35/ 2323
2F0_C384	Global Receive FIFO Size Register (USB1_GRXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	36.2.35/ 2323
2F0_C388	Global Receive FIFO Size Register (USB1_GRXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	36.2.35/ 2323
2F0_C38C	Global Receive FIFO Size Register (USB1_GRXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	36.2.35/ 2323
2F0_C400	Global Event Buffer Address (Low) Register (USB1_GEVNTADRLO)	32	R/W	0000_0000h	36.2.36/ 2323
2F0_C404	Global Event Buffer Address (High) Register (USB1_GEVNTADRHI)	32	R/W	0000_0000h	36.2.37/ 2324
2F0_C408	Global Event Buffer Size Register (USB1_GEVNTSIZ)	32	R/W	0000_0000h	36.2.38/ 2325
2F0_C40C	Global Event Buffer Count Register (USB1_GEVNTCOUNT)	32	R/W	0000_0000h	36.2.39/ 2326
2F0_C600	Global Hardware Parameters Register 8 (USB1_GHWPARAMS8)	32	R	<a href="#">See section</a>	36.2.40/ 2326

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2F0_C610	Global Device TX FIFO DMA Priority Register (USB1_GTXFIFOPRIDEV)	32	R/W	0000_0000h	<a href="#">36.2.41/ 2327</a>
2F0_C618	Global Host TX FIFO DMA Priority Register (USB1_GTXFIFOPRIHST)	32	R/W	0000_0000h	<a href="#">36.2.42/ 2328</a>
2F0_C61C	Global Host RX FIFO DMA Priority Register (USB1_GRXFIFOPRIHST)	32	R/W	0000_0000h	<a href="#">36.2.43/ 2329</a>
2F0_C624	Global Host FIFO DMA High-Low Priority Ratio Register (USB1_GDMAHLRATIO)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.44/ 2330</a>
2F0_C630	Global Frame Length Adjustment Register (USB1_GFLADJ)	32	R/W	0000_0000h	<a href="#">36.2.45/ 2331</a>
2F0_C700	Device Configuration Register (USB1_DCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.46/ 2332</a>
2F0_C704	Device Control Register (USB1_DCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.47/ 2334</a>
2F0_C708	Device Event Enable Register (USB1_DEVTEN)	32	R/W	0000_0000h	<a href="#">36.2.48/ 2339</a>
2F0_C70C	Device Status Register (USB1_DSTS)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.49/ 2341</a>
2F0_C710	Device Generic Command Parameter Register (USB1_DGCMMDPAR)	32	R/W	0000_0000h	<a href="#">36.2.50/ 2344</a>
2F0_C714	Device Generic Command Register (USB1_DGCMD)	32	R/W	0000_0000h	<a href="#">36.2.51/ 2344</a>
2F0_C720	Device Active USB Endpoint Enable Register (USB1_DALEPENA)	32	R/W	0000_0000h	<a href="#">36.2.52/ 2347</a>
2F0_C800	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_0)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C804	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_0)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C808	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_0)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C80C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_0)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C810	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_1)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C814	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_1)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C818	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_1)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C81C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_1)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C820	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_2)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C824	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_2)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>

Table continues on the next page...

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
2F0_C828	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_2)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C82C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_2)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C830	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_3)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C834	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_3)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C838	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_3)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C83C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_3)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C840	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_4)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C844	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_4)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C848	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_4)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C84C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_4)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C850	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_5)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C854	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_5)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C858	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_5)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C85C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_5)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C860	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_6)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C864	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_6)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C868	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_6)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C86C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_6)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
2F0_C870	Device Physical Endpoint-n Command Parameter 2 Register (USB1_DEPCMDPAR2_7)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
2F0_C874	Device Physical Endpoint-n Command Parameter 1 Register (USB1_DEPCMDPAR1_7)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
2F0_C878	Device Physical Endpoint-n Command Parameter 0 Register (USB1_DEPCMDPAR0_7)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
2F0_C87C	Device Physical Endpoint-n Command Register (USB1_DEPCMD_7)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2F0_CC00	OTG Configuration Register (USB1_OCFG)	32	R/W	0000_0000h	<a href="#">36.2.57/2352</a>
2F0_CC04	OTG Control Register (USB1_OCTL)	32	R/W	0000_0000h	<a href="#">36.2.58/2354</a>
2F0_CC08	OTG Events Register (USB1_OEVT)	32	R/W	0000_0000h	<a href="#">36.2.59/2357</a>
2F0_CC0C	OTG Events Enable Register (USB1_OEVTEM)	32	R/W	0000_0000h	<a href="#">36.2.60/2361</a>
2F0_CC10	OTG Status Register (USB1_OSTS)	32	R	<a href="#">See section</a>	<a href="#">36.2.61/2364</a>
2F0_CC20	ADP Configuration Register (USB1_ADPCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.62/2366</a>
2F0_CC24	ADP Control Register (USB1_ADPCCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.63/2367</a>
2F0_CC28	ADP Event Register (USB1_ADPEVT)	32	R/W	0000_0000h	<a href="#">36.2.64/2369</a>
2F0_CC2C	ADP Event Enable Register (USB1_ADPEVTEM)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.65/2371</a>
300_0000	Capability Registers Length and HC Interface Version Number (USB2_CAPLENGTH)	32	R	<a href="#">See section</a>	<a href="#">36.2.1/2268</a>
300_0004	Host Controller Structural Parameters 1 (USB2_HCSPARAMS1)	32	R	<a href="#">See section</a>	<a href="#">36.2.2/2269</a>
300_0008	Host Controller Structural Parameters 2 (USB2_HCSPARAMS2)	32	R	<a href="#">See section</a>	<a href="#">36.2.3/2269</a>
300_000C	Host Controller Structural Parameters 3 (USB2_HCSPARAMS3)	32	R	<a href="#">See section</a>	<a href="#">36.2.4/2270</a>
300_0010	Host Controller Capability Parameters 1 (USB2_HCCPARAMS1)	32	R	<a href="#">See section</a>	<a href="#">36.2.5/2271</a>
300_0014	Doorbell Offset (USB2_DBOFF)	32	R	<a href="#">See section</a>	<a href="#">36.2.6/2272</a>
300_0018	Runtime Register Space Offset (USB2_RTOSOFF)	32	R	<a href="#">See section</a>	<a href="#">36.2.7/2273</a>
300_001C	Host Controller Capability Parameters 2 (USB2_HCCPARAMS2)	32	R	<a href="#">See section</a>	<a href="#">36.2.8/2273</a>
300_C100	Global SoC Bus Configuration Register 0 (USB2_GSBUSCFG0)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.9/2274</a>
300_C104	Global SoC Bus Configuration Register 1 (USB2_GSBUSCFG1)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.10/2279</a>
300_C108	Global Tx Threshold Control Register (USB2_GTXTHRCFG)	32	R/W	0000_0000h	<a href="#">36.2.11/2280</a>
300_C10C	Global Rx Threshold Control Register (USB2_GRXTHRCFG)	32	R/W	0000_0000h	<a href="#">36.2.12/2281</a>
300_C110	Global Core Control Register (USB2_GCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.13/2283</a>
300_C118	Global Status Register (USB2_GSTS)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.14/2288</a>

Table continues on the next page...

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
300_C11C	Global User Control Register 1 (USB2_GUCTL1)	32	R/W	<a href="#">See section</a>	36.2.15/ 2290
300_C128	Global User ID Register (USB2_GUID)	32	R	<a href="#">See section</a>	36.2.16/ 2294
300_C12C	Global User Control Register (USB2_GUCTL)	32	R/W	<a href="#">See section</a>	36.2.17/ 2295
300_C130	Global SoC Bus Error Address Register low (USB2_GBUSERRADDRLO)	32	R	0000_0000h	36.2.18/ 2298
300_C134	Global SoC Bus Error Address Register high (USB2_GBUSERRADDRHI)	32	R	0000_0000h	36.2.19/ 2299
300_C138	Global SS Port to Bus Instance Mapping Register - Low (USB2_GPRTBIMAPLO)	32	R/W	0000_0000h	36.2.20/ 2299
300_C13C	Global SS Port to Bus Instance Mapping Register - High (USB2_GPRTBIMAPHI)	32	R	0000_0000h	36.2.21/ 2300
300_C140	Global Hardware Parameters Register 0 (USB2_GHWPARAMS0)	32	R	<a href="#">See section</a>	36.2.22/ 2300
300_C144	Global Hardware Parameters Register 1 (USB2_GHWPARAMS1)	32	R	<a href="#">See section</a>	36.2.23/ 2302
300_C148	Global Hardware Parameters Register 2 (USB2_GHWPARAMS2)	32	R	<a href="#">See section</a>	36.2.24/ 2305
300_C14C	Global Hardware Parameters Register 3 (USB2_GHWPARAMS3)	32	R	<a href="#">See section</a>	36.2.25/ 2306
300_C150	Global Hardware Parameters Register 4 (USB2_GHWPARAMS4)	32	R	<a href="#">See section</a>	36.2.26/ 2309
300_C154	Global Hardware Parameters Register 5 (USB2_GHWPARAMS5)	32	R	<a href="#">See section</a>	36.2.27/ 2311
300_C158	Global Hardware Parameters Register 6 (USB2_GHWPARAMS6)	32	R	<a href="#">See section</a>	36.2.28/ 2312
300_C15C	Global Hardware Parameters Register 7 (USB2_GHWPARAMS7)	32	R	<a href="#">See section</a>	36.2.29/ 2314
300_C180	Global High-Speed Port to Bus Instance Mapping Register - Low (USB2_GPRTBIMAP_HSLO)	32	R/W	0000_0000h	36.2.30/ 2315
300_C184	Global High-Speed Port to Bus Instance Mapping Register - High (USB2_GPRTBIMAP_HSHI)	32	R	0000_0000h	36.2.31/ 2315
300_C200	Global USB2 PHY Configuration Register (USB2_GUSB2PHYCFG)	32	R/W	<a href="#">See section</a>	36.2.32/ 2316
300_C2C0	Global USB 3.0 PIPE Control Register (USB2_GUSB3PIPECTL)	32	R/W	<a href="#">See section</a>	36.2.33/ 2319
300_C300	Global Transmit FIFO Size Register (USB2_GTXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
300_C304	Global Transmit FIFO Size Register (USB2_GTXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321
300_C308	Global Transmit FIFO Size Register (USB2_GTXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	36.2.34/ 2321

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
300_C30C	Global Transmit FIFO Size Register (USB2_GTXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.34/2321</a>
300_C380	Global Receive FIFO Size Register (USB2_GRXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
300_C384	Global Receive FIFO Size Register (USB2_GRXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
300_C388	Global Receive FIFO Size Register (USB2_GRXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
300_C38C	Global Receive FIFO Size Register (USB2_GRXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
300_C400	Global Event Buffer Address (Low) Register (USB2_GEVNTADRLO)	32	R/W	0000_0000h	<a href="#">36.2.36/2323</a>
300_C404	Global Event Buffer Address (High) Register (USB2_GEVNTADRHI)	32	R/W	0000_0000h	<a href="#">36.2.37/2324</a>
300_C408	Global Event Buffer Size Register (USB2_GEVNTSIZ)	32	R/W	0000_0000h	<a href="#">36.2.38/2325</a>
300_C40C	Global Event Buffer Count Register (USB2_GEVNTCOUNT)	32	R/W	0000_0000h	<a href="#">36.2.39/2326</a>
300_C600	Global Hardware Parameters Register 8 (USB2_GHWPARAMS8)	32	R	<a href="#">See section</a>	<a href="#">36.2.40/2326</a>
300_C610	Global Device TX FIFO DMA Priority Register (USB2_GTXFIFOPRIDEV)	32	R/W	0000_0000h	<a href="#">36.2.41/2327</a>
300_C618	Global Host TX FIFO DMA Priority Register (USB2_GTXFIFOPRIHOST)	32	R/W	0000_0000h	<a href="#">36.2.42/2328</a>
300_C61C	Global Host RX FIFO DMA Priority Register (USB2_GRXFIFOPRIHOST)	32	R/W	0000_0000h	<a href="#">36.2.43/2329</a>
300_C624	Global Host FIFO DMA High-Low Priority Ratio Register (USB2_GDMAHLRATIO)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.44/2330</a>
300_C630	Global Frame Length Adjustment Register (USB2_GFLADJ)	32	R/W	0000_0000h	<a href="#">36.2.45/2331</a>
300_C700	Device Configuration Register (USB2_DCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.46/2332</a>
300_C704	Device Control Register (USB2_DCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.47/2334</a>
300_C708	Device Event Enable Register (USB2_DEVTEN)	32	R/W	0000_0000h	<a href="#">36.2.48/2339</a>
300_C70C	Device Status Register (USB2_DSTS)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.49/2341</a>
300_C710	Device Generic Command Parameter Register (USB2_DGCMMDPAR)	32	R/W	0000_0000h	<a href="#">36.2.50/2344</a>
300_C714	Device Generic Command Register (USB2_DGCMMD)	32	R/W	0000_0000h	<a href="#">36.2.51/2344</a>
300_C720	Device Active USB Endpoint Enable Register (USB2_DALEPENA)	32	R/W	0000_0000h	<a href="#">36.2.52/2347</a>

*Table continues on the next page...*

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
300_C800	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_0)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C804	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_0)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
300_C808	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_0)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
300_C80C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_0)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
300_C810	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_1)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C814	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_1)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
300_C818	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_1)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
300_C81C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_1)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
300_C820	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_2)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C824	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_2)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
300_C828	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_2)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
300_C82C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_2)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
300_C830	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_3)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C834	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_3)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
300_C838	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_3)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
300_C83C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_3)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
300_C840	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_4)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C844	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_4)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
300_C848	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_4)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
300_C84C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_4)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
300_C850	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_5)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
300_C854	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_5)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
300_C858	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_5)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
300_C85C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_5)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>
300_C860	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_6)	32	R/W	0000_0000h	<a href="#">36.2.53/2348</a>
300_C864	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_6)	32	R/W	0000_0000h	<a href="#">36.2.54/2349</a>
300_C868	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_6)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
300_C86C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_6)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>
300_C870	Device Physical Endpoint-n Command Parameter 2 Register (USB2_DEPCMDPAR2_7)	32	R/W	0000_0000h	<a href="#">36.2.53/2348</a>
300_C874	Device Physical Endpoint-n Command Parameter 1 Register (USB2_DEPCMDPAR1_7)	32	R/W	0000_0000h	<a href="#">36.2.54/2349</a>
300_C878	Device Physical Endpoint-n Command Parameter 0 Register (USB2_DEPCMDPAR0_7)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
300_C87C	Device Physical Endpoint-n Command Register (USB2_DEPCMD_7)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>
300_CC00	OTG Configuration Register (USB2_OCFG)	32	R/W	0000_0000h	<a href="#">36.2.57/2352</a>
300_CC04	OTG Control Register (USB2_OCTL)	32	R/W	0000_0000h	<a href="#">36.2.58/2354</a>
300_CC08	OTG Events Register (USB2_OEVT)	32	R/W	0000_0000h	<a href="#">36.2.59/2357</a>
300_CC0C	OTG Events Enable Register (USB2_OEVTEM)	32	R/W	0000_0000h	<a href="#">36.2.60/2361</a>
300_CC10	OTG Status Register (USB2_OSTS)	32	R	<a href="#">See section</a>	<a href="#">36.2.61/2364</a>
300_CC20	ADP Configuration Register (USB2_ADPCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.62/2366</a>
300_CC24	ADP Control Register (USB2_ADPCCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.63/2367</a>
300_CC28	ADP Event Register (USB2_ADPEVT)	32	R/W	0000_0000h	<a href="#">36.2.64/2369</a>
300_CC2C	ADP Event Enable Register (USB2_ADPEVTEM)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.65/2371</a>
310_0000	Capability Registers Length and HC Interface Version Number (USB3_CAPLENGTH)	32	R	<a href="#">See section</a>	<a href="#">36.2.1/2268</a>
310_0004	Host Controller Structural Parameters 1 (USB3_HCSPARAMS1)	32	R	<a href="#">See section</a>	<a href="#">36.2.2/2269</a>
310_0008	Host Controller Structural Parameters 2 (USB3_HCSPARAMS2)	32	R	<a href="#">See section</a>	<a href="#">36.2.3/2269</a>

*Table continues on the next page...*

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
310_000C	Host Controller Structural Parameters 3 (USB3_HCSPARAMS3)	32	R	<a href="#">See section</a>	36.2.4/2270
310_0010	Host Controller Capability Parameters 1 (USB3_HCCPARAMS1)	32	R	<a href="#">See section</a>	36.2.5/2271
310_0014	Doorbell Offset (USB3_DBOFF)	32	R	<a href="#">See section</a>	36.2.6/2272
310_0018	Runtime Register Space Offset (USB3_RTSOFF)	32	R	<a href="#">See section</a>	36.2.7/2273
310_001C	Host Controller Capability Parameters 2 (USB3_HCCPARAMS2)	32	R	<a href="#">See section</a>	36.2.8/2273
310_C100	Global SoC Bus Configuration Register 0 (USB3_GSBUSCFG0)	32	R/W	<a href="#">See section</a>	36.2.9/2274
310_C104	Global SoC Bus Configuration Register 1 (USB3_GSBUSCFG1)	32	R/W	<a href="#">See section</a>	36.2.10/2279
310_C108	Global Tx Threshold Control Register (USB3_GTXTHRCFG)	32	R/W	0000_0000h	36.2.11/2280
310_C10C	Global Rx Threshold Control Register (USB3_GRXTHRCFG)	32	R/W	0000_0000h	36.2.12/2281
310_C110	Global Core Control Register (USB3_GCTL)	32	R/W	<a href="#">See section</a>	36.2.13/2283
310_C118	Global Status Register (USB3_GSTS)	32	R/W	<a href="#">See section</a>	36.2.14/2288
310_C11C	Global User Control Register 1 (USB3_GUCTL1)	32	R/W	<a href="#">See section</a>	36.2.15/2290
310_C128	Global User ID Register (USB3_GUID)	32	R	<a href="#">See section</a>	36.2.16/2294
310_C12C	Global User Control Register (USB3_GUCTL)	32	R/W	<a href="#">See section</a>	36.2.17/2295
310_C130	Global SoC Bus Error Address Register low (USB3_GBUSERRADDRLO)	32	R	0000_0000h	36.2.18/2298
310_C134	Global SoC Bus Error Address Register high (USB3_GBUSERRADDRHI)	32	R	0000_0000h	36.2.19/2299
310_C138	Global SS Port to Bus Instance Mapping Register - Low (USB3_GPRTBIMAPLO)	32	R/W	0000_0000h	36.2.20/2299
310_C13C	Global SS Port to Bus Instance Mapping Register - High (USB3_GPRTBIMAPHI)	32	R	0000_0000h	36.2.21/2300
310_C140	Global Hardware Parameters Register 0 (USB3_GHWPARAMS0)	32	R	<a href="#">See section</a>	36.2.22/2300
310_C144	Global Hardware Parameters Register 1 (USB3_GHWPARAMS1)	32	R	<a href="#">See section</a>	36.2.23/2302
310_C148	Global Hardware Parameters Register 2 (USB3_GHWPARAMS2)	32	R	<a href="#">See section</a>	36.2.24/2305
310_C14C	Global Hardware Parameters Register 3 (USB3_GHWPARAMS3)	32	R	<a href="#">See section</a>	36.2.25/2306
310_C150	Global Hardware Parameters Register 4 (USB3_GHWPARAMS4)	32	R	<a href="#">See section</a>	36.2.26/2309

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
310_C154	Global Hardware Parameters Register 5 (USB3_GHWPARAMS5)	32	R	<a href="#">See section</a>	<a href="#">36.2.27/2311</a>
310_C158	Global Hardware Parameters Register 6 (USB3_GHWPARAMS6)	32	R	<a href="#">See section</a>	<a href="#">36.2.28/2312</a>
310_C15C	Global Hardware Parameters Register 7 (USB3_GHWPARAMS7)	32	R	<a href="#">See section</a>	<a href="#">36.2.29/2314</a>
310_C180	Global High-Speed Port to Bus Instance Mapping Register - Low (USB3_GPRTBIMAP_HSLO)	32	R/W	0000_0000h	<a href="#">36.2.30/2315</a>
310_C184	Global High-Speed Port to Bus Instance Mapping Register - High (USB3_GPRTBIMAP_HSHI)	32	R	0000_0000h	<a href="#">36.2.31/2315</a>
310_C200	Global USB2 PHY Configuration Register (USB3_GUSB2PHYCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.32/2316</a>
310_C2C0	Global USB 3.0 PIPE Control Register (USB3_GUSB3PIPECTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.33/2319</a>
310_C300	Global Transmit FIFO Size Register (USB3_GTXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.34/2321</a>
310_C304	Global Transmit FIFO Size Register (USB3_GTXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.34/2321</a>
310_C308	Global Transmit FIFO Size Register (USB3_GTXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.34/2321</a>
310_C30C	Global Transmit FIFO Size Register (USB3_GTXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.34/2321</a>
310_C380	Global Receive FIFO Size Register (USB3_GRXFIFOSIZ_0)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
310_C384	Global Receive FIFO Size Register (USB3_GRXFIFOSIZ_1)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
310_C388	Global Receive FIFO Size Register (USB3_GRXFIFOSIZ_2)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
310_C38C	Global Receive FIFO Size Register (USB3_GRXFIFOSIZ_3)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.35/2323</a>
310_C400	Global Event Buffer Address (Low) Register (USB3_GEVNTADRLO)	32	R/W	0000_0000h	<a href="#">36.2.36/2323</a>
310_C404	Global Event Buffer Address (High) Register (USB3_GEVNTADRHI)	32	R/W	0000_0000h	<a href="#">36.2.37/2324</a>
310_C408	Global Event Buffer Size Register (USB3_GEVNTSIZ)	32	R/W	0000_0000h	<a href="#">36.2.38/2325</a>
310_C40C	Global Event Buffer Count Register (USB3_GEVNTCOUNT)	32	R/W	0000_0000h	<a href="#">36.2.39/2326</a>
310_C600	Global Hardware Parameters Register 8 (USB3_GHWPARAMS8)	32	R	<a href="#">See section</a>	<a href="#">36.2.40/2326</a>
310_C610	Global Device TX FIFO DMA Priority Register (USB3_GTXFIFOPRIDEV)	32	R/W	0000_0000h	<a href="#">36.2.41/2327</a>
310_C618	Global Host TX FIFO DMA Priority Register (USB3_GTXFIFOPRIHOST)	32	R/W	0000_0000h	<a href="#">36.2.42/2328</a>

*Table continues on the next page...*

## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
310_C61C	Global Host RX FIFO DMA Priority Register (USB3_GRXFIFOPRIHST)	32	R/W	0000_0000h	<a href="#">36.2.43/2329</a>
310_C624	Global Host FIFO DMA High-Low Priority Ratio Register (USB3_GDMAHLRATIO)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.44/2330</a>
310_C630	Global Frame Length Adjustment Register (USB3_GFLADJ)	32	R/W	0000_0000h	<a href="#">36.2.45/2331</a>
310_C700	Device Configuration Register (USB3_DCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.46/2332</a>
310_C704	Device Control Register (USB3_DCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.47/2334</a>
310_C708	Device Event Enable Register (USB3_DEVTEN)	32	R/W	0000_0000h	<a href="#">36.2.48/2339</a>
310_C70C	Device Status Register (USB3_DSTS)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.49/2341</a>
310_C710	Device Generic Command Parameter Register (USB3_DGCMMDPAR)	32	R/W	0000_0000h	<a href="#">36.2.50/2344</a>
310_C714	Device Generic Command Register (USB3_DGCMD)	32	R/W	0000_0000h	<a href="#">36.2.51/2344</a>
310_C720	Device Active USB Endpoint Enable Register (USB3_DALEPENA)	32	R/W	0000_0000h	<a href="#">36.2.52/2347</a>
310_C800	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_0)	32	R/W	0000_0000h	<a href="#">36.2.53/2348</a>
310_C804	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_0)	32	R/W	0000_0000h	<a href="#">36.2.54/2349</a>
310_C808	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_0)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
310_C80C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_0)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>
310_C810	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_1)	32	R/W	0000_0000h	<a href="#">36.2.53/2348</a>
310_C814	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_1)	32	R/W	0000_0000h	<a href="#">36.2.54/2349</a>
310_C818	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_1)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
310_C81C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_1)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>
310_C820	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_2)	32	R/W	0000_0000h	<a href="#">36.2.53/2348</a>
310_C824	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_2)	32	R/W	0000_0000h	<a href="#">36.2.54/2349</a>
310_C828	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_2)	32	R/W	0000_0000h	<a href="#">36.2.55/2349</a>
310_C82C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_2)	32	R/W	0000_0000h	<a href="#">36.2.56/2350</a>

Table continues on the next page...

**USB memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
310_C830	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_3)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
310_C834	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_3)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
310_C838	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_3)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
310_C83C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_3)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
310_C840	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_4)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
310_C844	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_4)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
310_C848	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_4)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
310_C84C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_4)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
310_C850	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_5)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
310_C854	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_5)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
310_C858	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_5)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
310_C85C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_5)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
310_C860	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_6)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
310_C864	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_6)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
310_C868	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_6)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
310_C86C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_6)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
310_C870	Device Physical Endpoint-n Command Parameter 2 Register (USB3_DEPCMDPAR2_7)	32	R/W	0000_0000h	<a href="#">36.2.53/ 2348</a>
310_C874	Device Physical Endpoint-n Command Parameter 1 Register (USB3_DEPCMDPAR1_7)	32	R/W	0000_0000h	<a href="#">36.2.54/ 2349</a>
310_C878	Device Physical Endpoint-n Command Parameter 0 Register (USB3_DEPCMDPAR0_7)	32	R/W	0000_0000h	<a href="#">36.2.55/ 2349</a>
310_C87C	Device Physical Endpoint-n Command Register (USB3_DEPCMD_7)	32	R/W	0000_0000h	<a href="#">36.2.56/ 2350</a>
310_CC00	OTG Configuration Register (USB3_OCFG)	32	R/W	0000_0000h	<a href="#">36.2.57/ 2352</a>
310_CC04	OTG Control Register (USB3_OCTL)	32	R/W	0000_0000h	<a href="#">36.2.58/ 2354</a>

*Table continues on the next page...*

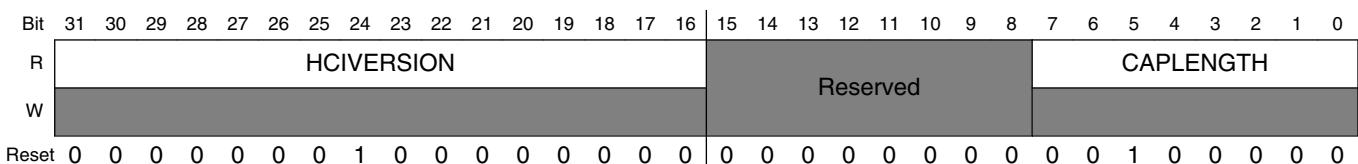
## USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
310_CC08	OTG Events Register (USB3_OEVT)	32	R/W	0000_0000h	<a href="#">36.2.59/2357</a>
310_CC0C	OTG Events Enable Register (USB3_OEVTC)	32	R/W	0000_0000h	<a href="#">36.2.60/2361</a>
310_CC10	OTG Status Register (USB3_OSTS)	32	R	<a href="#">See section</a>	<a href="#">36.2.61/2364</a>
310_CC20	ADP Configuration Register (USB3_ADPCFG)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.62/2366</a>
310_CC24	ADP Control Register (USB3_ADPCCTL)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.63/2367</a>
310_CC28	ADP Event Register (USB3_ADPEVT)	32	R/W	0000_0000h	<a href="#">36.2.64/2369</a>
310_CC2C	ADP Event Enable Register (USB3_ADPEVTC)	32	R/W	<a href="#">See section</a>	<a href="#">36.2.65/2371</a>

### 36.2.1 Capability Registers Length and HC Interface Version Number (USBx\_CAPLENGTH)

The register is read-only.

Address: Base address + 0h offset



#### USBx\_CAPLENGTH field descriptions

Field	Description
31–16 HCIVERSION	HC Interface Version Number. The value is set as 32'h100.
15–8 —	This field is reserved. -
CAPLENGTH	Capability registers length. The value is 32'h20.

### 36.2.2 Host Controller Structural Parameters 1 (USBx\_HCSPARAMS1)

The register is read-only.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAXPORTS										Reserved										MAXINTRS											
W																					MAXSLOTS											
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	

#### USBx\_HCSPARAMS1 field descriptions

Field	Description																												
31–24 MAXPORTS	Maximum number of ports, set as 2.																												
23–19 —	This field is reserved.																												
18–8 MAXINTRS	Number of Interrupters, set as 1.																												
MAXSLOTS	Number of Device Slots set as 127.																												

### 36.2.3 Host Controller Structural Parameters 2 (USBx\_HCSPARAMS2)

The register is read-only.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16														
R	MAXSCRATCHPADBUFS										SPR	MAXSCRATCHPADBUFS_HI										Reserved								
W																														
Reset	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	ERSTMAX										IST			
R	Reserved																													
W																														
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

#### USBx\_HCSPARAMS2 field descriptions

Field	Description																												
31–27 MAXSCRATCHPADBUFS	Max Scratchpad Bufs Low:2.																												

Table continues on the next page...

**USBx\_HCSPARAMS2 field descriptions (continued)**

Field	Description
26 SPR	Scratchpad Restore
25–21 MAXSCRATCHPADBUFS_HI	Max Scratchpad Bufs High
20–8 —	This field is reserved.
7–4 ERSTMAX	Event ring segment table max, 15
IST	Isochronous Scheduling Threshold, set as 1.

**36.2.4 Host Controller Structural Parameters 3 (USBx\_HCSPARAMS3)**

The register is read-only.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
	U2_DEVICE_EXIT_LAT																Reserved					U1_DEVICE_EXIT_LAT										
W																																
Reset	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

**USBx\_HCSPARAMS3 field descriptions**

Field	Description
31–16 U2_DEVICE_EXIT_LAT	U2 Device Exit Latency, set as 32'h7ff.
15–8 —	This field is reserved.
U1_DEVICE_EXIT_LAT	U1 Device Exit Latency, set as 32'ha

### 36.2.5 Host Controller Capability Parameters 1 (USBx\_HCCPARAMS1)

The register is read-only.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	xECP															
W																
Reset	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAXPSASIZE				CFC	SEC	SPC	PAE	NSS	LTC	LHRC	PIND	PPC	CSZ	BNC	AC64
W																
Reset	1	1	1	1	0	1	1	0	0	1	1	0	1	1	0	1

#### USBx\_HCCPARAMS1 field descriptions

Field	Description
31–16 xECP	xHCI Extended Capabilities Pointer, set as 544.
15–12 MAXPSASIZE	Maximum Primary Stream Array Size, set as 15
11 CFC	Contiguous Frame ID Capability
10 SEC	Stopped EDLTA Capability
9 SPC	Short Packet Capability
8 PAE	Parse All Event Data
7 NSS	No Secondary SID Support

Table continues on the next page...

**USBx\_HCCPARAMS1 field descriptions (continued)**

Field	Description
6 LTC	Latency Tolerance Messaging Capability
5 LHRC	Light HC Reset Capability
4 PIND	Port Indicators
3 PPC	Port Power Control
2 CSZ	Context Size
1 BNC	BW Negotiation Capability
0 AC64	64-bit Addressing Capability

**36.2.6 Doorbell Offset (USBx\_DBOFF)**

The register is read-only.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DOORBELL_ARRAY_OFFSET															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DOORBELL_ARRAY_OFFSET												Reserved			
W																
Reset	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0

**USBx\_DBOFF field descriptions**

Field	Description
31–2 DOORBELL_ARRAY_OFFSET	Doorbell Array Offset, set as 1152.
—	This field is reserved.

### 36.2.7 Runtime Register Space Offset (USBx\_RT\_OFFSET)

The register is read-only.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																	
W																																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0		

#### USBx\_RT\_OFFSET field descriptions

Field	Description																												
31–5 RUNTIME_REG_SPACE_OFFSET	Runtime Register Space Offset, set as 1088.																												
—	This field is reserved.																												

### 36.2.8 Host Controller Capability Parameters 2 (USBx\_HCCPARAMS2)

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16																
R																																	
W																																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0																
Bit	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0		CIC	LEC	CTC	FSC	CMC	U3C									
R																																	
W																																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0		0	0	0	1	0	1	0	1	1	1					

#### USBx\_HCCPARAMS2 field descriptions

Field	Description																													
31–6 —	This field is reserved.																													
5 CIC	Configuration Information Capability																													
4 LEC	Large ESIT Payload Capability																													
3 CTC	Compliance Transition Capability																													

Table continues on the next page...

**USBx\_HCCPARAMS2 field descriptions (continued)**

Field	Description
2 FSC	Force Save Context Capability
1 CMC	Configure Endpoint Command Max Exit Latency Too Large Capability
0 U3C	U3 Entry Capability

**36.2.9 Global SoC Bus Configuration Register 0 (USBx\_GSBUSCFG0)**

This register configures system bus DMA options for the AXI master bus. Options include burst length and cache type (bufferable/posted, cacheable/snoop, and so on). The application can program this register upon power-on, or a change in mode of operation after the DMA engine is halted.

**xHCI Register Power-On Value**

If you are using a standard xHCI host driver, make sure to set the register's power-on value (Current values are 32'h100080 and 32'h700) because the standard xHCI driver does not access this register.

**Burst Length**

For a given DMA transfer, the burst length is set according to the largest enabled burst length. Note that the Undefined Length INCR burst, if enabled, has priority over all other burst lengths.

**Cache Type**

For a given DMA transfer, the cache type is set according to the 4-bit cache type register field corresponding to the transfer type:

- Data read
- Descriptor read
- Data write
- Descriptor write

The definition of the 4-bit cache type register field corresponds to the cache type definition of the configured master bus type (AXI3) as defined below:

**Table 36-4. Cache Type Bit Assignments**

MBUS_TYPE	Cache Type Output	[3]	[2]	[1]	[0]
AXI3	ar/awcache[3:0]	Write Allocate	Read Allocate	Cacheable	Bufferable

## Cache Type in /AXI

AXI3: CSR cache type refers to AXI cache type (awcache[3:0] and arcache[3:0]); or in ACE (AXI coherency extensions) to AXI memory type.

## Posted Requests

The native interface DMA read request gmr\_mcmd[2:0] indicates posted vs. non-posted requests depending on the bufferable/posted bit of the cache type.

**Table 36-5. Cache type bit definitions and usage**

Signal Name	Definition	Usage
Bufferable, Posted	The transaction response can be returned quickly, but the transaction itself can take an arbitrary number of clock cycles to reach the final destination.	<ul style="list-style-type: none"> <li>Improves bus utilization of DMA writes with a high response latency, such as off-chip memory or bridges.</li> <li>Allows the core to start a new DMA write transaction while a DMA write is in progress.</li> <li>For descriptor write transactions, "Bufferable/Posted" must be zero to avoid a DMA write race condition with a software interrupt or subsequent DMA read. When the core writes back a descriptor or event, it waits for the DMA response to indicate that it can safely re-fetch that descriptor or set the core's interrupt. If the descriptor DMA write is posted, the data integrity is not ensured for future transactions.</li> </ul>
Cacheable, Modifiable, Snoop (negation of No Snoop)	The characteristics of the transaction at the destination may not match the original.	<ul style="list-style-type: none"> <li>For DMA writes, multiple write transactions may be merged.</li> <li>For DMA reads, a location can be pre-fetched or fetched just once for multiple read transactions.</li> <li>For AXI, this bit should be used in conjunction with the Read Allocate / Write Allocate to indicate upon a cache miss whether the transaction should be cached, or Other Allocate bits to indicate how to handle cache misses.</li> </ul>

## Connections from GSBUSCFG0 to Master Cache Type

In device mode, the GSBUSCFG0 cache type fields are connected to the two (write and read) system bus master cache type ports through a pair of 4-bit wide 2x1 multiplexers; the MUX select signal is the descriptor access indication (as opposed to data).

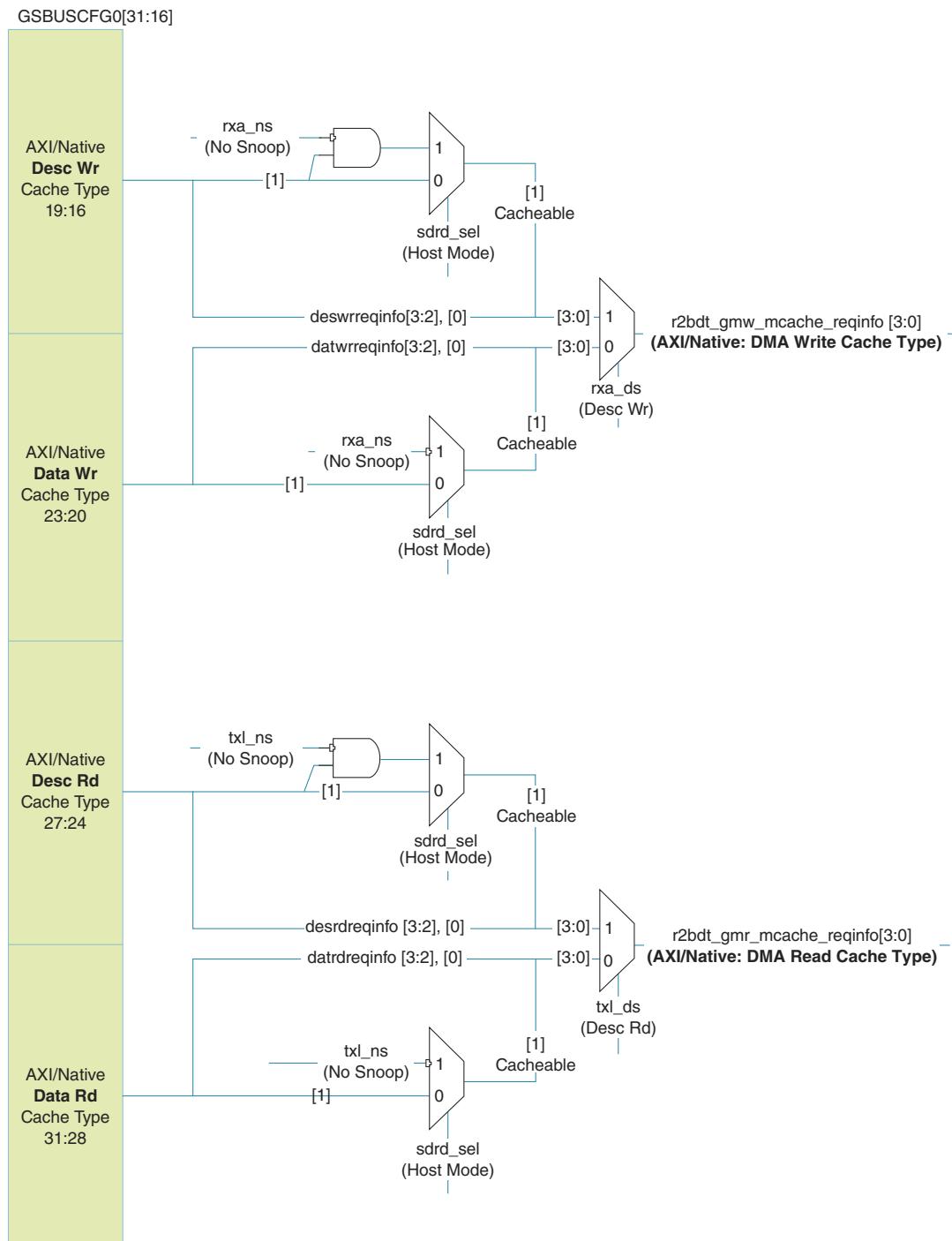
In host mode, the GSBUSCFG0 cache type fields are connected as in device mode with these exceptions:

- For data accesses, the core sets Cacheable to the inverse of the xHCI TRB's (Transfer request block) No Snoop flag; the Cacheable bit in GSBUSCFG0 is disregarded.
- For descriptor accesses, the core sets Cacheable to the inverse of the internal DMA command's No Snoop flag. When the LSP performs a Scratchpad DMA write, it sets

## USB Memory Map/Register Definition

the internal DMA command's No Snoop flag thereby ensuring Cacheable=0, as required by the xHCI specification.

The master bus type is AXI, the GSBUSCFG0 to Cache Type connections are reflected by following diagrams.



**Figure 36-2. GSBUSCFG0 Cache Type Connections for AXI**

**NOTE**

For master interface DMA access, program the GSBUSCFG0 register to 0x2222000F for better performance.

Address: Base address + C100h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATRDREQINFO				DESRDREQINFO				DATWRREQINFO				DESWRREQINFO			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								INCR256BRSTE NA	INCR128BRSTE NA	INCR64BRSTEN A	INCR32BRSTEN A	INCR16BRSTEN A	INCR8BRSTENA	INCR4BRSTENA	INCRBRSTENA
W									1	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

**USBx\_GSBUSCFG0 field descriptions**

Field	Description
31–28 DATRDREQINFO	AXI-cache for Data Read Input to BUS-GM.
27–24 DESRDREQINFO	AXI-cache for Descriptor Read Input to BUS-GM.
23–20 DATWRREQINFO	AXI-cache for Data Write Input to BUS-GM.
19–16 DESWRREQINFO	AXI-cache for descriptor write Input to BUS-GM.
15–8 —	This field is reserved.
7 INCR256BRSTENA	NCR256 Burst Type Enable Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 256-beat burst.  <b>NOTE:</b> The bit is not supported for now.
6 INCR128BRSTENA	NCR128 Burst Type Enable Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 128-beat burst.  <b>NOTE:</b> The bit is not supported for now.
5 INCR64BRSTENA	INCR64 Burst Type Enable Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 64-beat burst.

Table continues on the next page...

**USBx\_GSBUSCFG0 field descriptions (continued)**

Field	Description
	<b>NOTE:</b> The bit is not supported for now.
4 INCR32BRSTENA	NCR32 Burst Type Enable  Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 32-beat burst.  <b>NOTE:</b> The bit is not supported for now.
3 INCR16BRSTENA	INCR16 Burst Type Enable  Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 16-beat burst.
2 INCR8BRSTENA	INCR8 Burst Type Enable  Input to BUS-GM. For the AXI configuration, if software set this bit to "1", the AXI master uses INCR to do the 8-beat burst.
1 INCR4BRSTENA	INCR4 Burst Type Enable  Input to BUS-GM.  For the AXI configuration, when this bit is enabled the controller is allowed to do bursts of beat length 1, 2, 3, and 4. It is highly recommended that this bit is enabled to prevent descriptor reads and writes from being broken up into separate transfers.
0 INCRBRSTENA	Undefined Length INCR Burst Type Enable  Input to BUS-GM  This bit determines the set of burst lengths the master interface uses. It works in conjunction with the GSBUSCFG0[7:1] enables (INCR256/128/64/32/16/8/4).  0 INCRX burst mode  ARLEN/AWLEN (for AXI configurations) use only the following burst lengths:  1 (if GSBUSCFG0[1:7] = 0) 4 (if GSBUSCFG0[INCR4BrstEna] = 1) 8 (if GSBUSCFG0[INCR8BrstEna] = 1) 16 (if GSBUSCFG0[INCR16BrstEna] = 1) 32 (if GSBUSCFG0[INCR32BrstEna] = 1) 64 (if GSBUSCFG0[INCR64BrstEna] = 1) 128 (if GSBUSCFG0[INCR128BrstEna] = 1) 256 (if GSBUSCFG0[INCR256BrstEna] = 1)  They do not use INCR.  1 INCR (undefined length) burst mode  AXI configurations- ARLEN/AWLEN uses any length less than or equal to the largest-enabled burst length of INCR4/8/16/32/64/128/256.  For cache line-aligned applications, this bit is typically set to 0 to ensure that the master interface uses only power-of-2 burst lengths (as enabled via GSBUSCFG0[7:0]).

### 36.2.10 Global SoC Bus Configuration Register 1 (USBx\_GSBUSCFG1)

#### NOTE

For master interface DMA access, program the GSBUSCFG1 register to 0x00000F00 for better performance.

Address: Base address + C104h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

#### USBx\_GSBUSCFG1 field descriptions

Field	Description
31–13 —	This field is reserved.
12 EN1KPAGE	1k Page Boundary Enable By default (this bit is disabled) the AXI breaks transfers at the 4k page boundary. When this bit is enabled, the AXI master (DMA data) breaks transfers at the 1k page boundary.
11–8 PIPETRANSLIMIT	AXI Pipelined Transfers Burst Request Limit The field controls the number of outstanding pipelined transfers requests the AXI master pushes to the AXI slave. Once the AXI master reaches this limit, it does not make more requests on the AXI ARADDR and AWADDR buses until the associated data phases complete.  This field is encoded as follows- 0000- 1 request 0001- 2 requests 0010- 3 requests 0011- 4 requests ... 1111- 16 requests
—	This field is reserved.

### 36.2.11 Global Tx Threshold Control Register (USBx\_GTXTHRCFG)

All the fields in GTXTHRCFG register are valid only in Host mode. GTXTHRCFG register is not applicable for Debug Target. GTXTHRCFG register is not applicable in USB 2.0-only mode.

Address: Base address + C108h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
	Reserved		USBTxPktCntSel		Reserved		USBTxPktCnt						USBMaxTxBurstSize			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R													Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### USBx\_GTXTHRCFG field descriptions

Field	Description
31–30 —	This field is reserved.
29 USBTxPktCntSel	USB Transmit Packet Count Enable This field enables/disables the USB transmission multi-packet thresholding. 0 USB transmission multi-packet thresholding is disabled, the core can only start transmission on the USB after the entire packet has been fetched into the corresponding TXFIFO. 1 USB transmission multi-packet thresholding is enabled. The core can only start transmission on the USB after USB Transmit Packet Count amount of packets for the USB transaction (burst) are already in the corresponding TXFIFO. This mode is only valid in the host mode. It is only used for SuperSpeed.
28 —	This field is reserved.

Table continues on the next page...

**USBx\_GTXTHRCFG field descriptions (continued)**

Field	Description
27–24 USBTxPktCnt	USB Transmit Packet Count  This field specifies the number of packets that must be in the TXFIFO before the core can start transmission for the corresponding USB transaction (burst). This field is only valid when the USB Transmit Packet Count Enable field is set to one. Valid values are from 1 to 15.  <b>NOTE:</b> This field must be less than or equal to the USB Maximum TX Burst Size field.
23–16 USBMaxTxBurstSize	USB Maximum TX Burst Size  When USBTxPktCntSel is one, this field specifies the Maximum Bulk OUT burst the core can do. When the system bus is slower than the USB, TX FIFO can underrun during a long burst. User can program a smaller value to this field to limit the TX burst size that the core can do. It only applies to SS Bulk, Isochronous, and Interrupt OUT endpoints in the host mode. Valid values are from 1 to 16.
—	This field is reserved.

**36.2.12 Global Rx Threshold Control Register (USBx\_GRXTHRCFG)**

GRXTHRCFG register is not applicable for Debug Target. GRXTHRCFG register is not applicable in USB 2.0-only mode.

Address: Base address + C10Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		USBRxPktCntSel	Reserved												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBx\_GRXTHRCFG field descriptions**

Field	Description
31–30 —	This field is reserved.
29 USBRxPktCntSel	<p>USB ReceivePacket Count Enable</p> <p>This field enables/disables the USB reception multi-packet thresholding-</p> <p>0 The core can only start reception on the USB when the RX FIFO has space for at least one packet.</p> <p>1 The core can only start reception on the USB when the RX FIFO has space for at least USBRxPktCnt amount of packets. This mode is valid in both host and device mode. It is only used for SuperSpeed.</p> <p>In device mode,</p> <p>Setting this bit to 1 also enables the functionality of reporting NUMP in the ACK TP based on the RX FIFO space instead of reporting a fixed NUMP derived from DCFG[NUMP]</p> <p>If you are using external buffer control (EBC) feature, disable this mode by setting USBRxPktCntSel to 0</p>
28 —	This field is reserved.
27–24 USBRxPktCnt	<p>USB Receive Packet Count</p> <p>In host mode, this field specifies the space (in terms of the number of packets) that must be available in the RX FIFO before the core can start the corresponding USB RX transaction (burst).</p> <p>In device mode, this field specifies the space (in terms of the number of packets) that must be available in the RX FIFO before the core can send ERDY for a flow-controlled endpoint.</p> <p>This field is valid only when the USB Receive Packet Count Enable field is set to one. The valid values for this field are from 1 to 15.</p> <p><b>NOTE:</b> This field must be less than or equal to the USB Maximum Receive Burst Size field.</p>
23–19 USBMaxRxBurstSize	<p>USB Maximum Receive Burst Size</p> <p>In host mode, this field specifies the Maximum Bulk IN burst the USB 3.0 core can perform. When the system bus is slower than the USB, RX FIFO can overrun during a long burst. User can program a smaller value to this field to limit the RX burst size that the core can perform. It only applies to SS Bulk, Isochronous, and Interrupt IN endpoints in the host mode.</p> <p>In device mode, this field specifies the NUMP value that will be sent in ERDY for an OUT endpoint.</p> <p>This field is valid only when USBRxPktCntSel is one. The valid values for this field are from 1 to 16.</p>
—	This field is reserved.

### 36.2.13 Global Core Control Register (USBx\_GCTL)

Address: Base address + C110h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
PWRDNSCALE																
W																
Reset	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R						SOFITPSYNC											
	FRMSCLDWN		PRTCAPDIR	CORESOFTRESET			Reserved	Reserved			RAMCLKSEL		Reserved	DISSCRAMBLE	U2EXIT_LFPS	Reserved	
W																DSBLCLKGTNG	
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0

**USBx\_GCTL field descriptions**

Field	Description
31–19 PWRDNSCALE	<p>Power Down Scale</p> <p>The USB3 suspend_clk input replaces pipe3_rx_pclk as a clock source to a small part of the USB3 core that operates when the SS PHY is in its lowest power (P3) state, and therefore does not provide a clock.</p> <p>The Power Down Scale field specifies how many suspend_clk periods fit into a 16 kHz clock period. When performing the division, round up the remainder.</p> <p>For example, when using an 8-bit/16-bit/32-bit PHY and 25-MHz Suspend clock,</p> <p>Power Down Scale = <math>25000 \text{ kHz} / 16 \text{ kHz} = 13'd1563</math> (rounder up)</p> <p>Note-</p> <p>Minimum Suspend clock frequency is 32 kHz</p> <p>Maximum Suspend clock frequency is 125 MHz</p> <p>The LTSSM uses Suspend clock for 12-ms and 100-ms timers during suspend mode. According to the USB 3.0 specification, the accuracy on these timers is 0% to +50%.</p> <p><math>12 \text{ ms} + 0\text{--}+50\% \text{ accuracy} = 18 \text{ ms}</math> (Range is 12 ms - 18 ms)</p> <p><math>100 \text{ ms} + 0\text{--}+50\% \text{ accuracy} = 150 \text{ ms}</math> (Range is 100 ms - 150 ms).</p> <p>The suspend clock accuracy requirement is-</p> <p><math>(12,000/62.5) * (\text{GCTL}[31:19]) * \text{actual suspend\_clk\_period}</math> should be between 12,000 and 18,000</p>

*Table continues on the next page...*

**USBx\_GCTL field descriptions (continued)**

Field	Description
	(100,000/62.5) * (GCTL[31:19]) * actual suspend_clk_period should be between 100,000 and 150,000 For example, if your suspend_clk frequency varies from 7.5 MHz to 10.5MHz, then the value needs to programmed is- Power Down Scale = 10500/16 = 657 (rounded up; and fastest frequency used)
18 MASTERFILTBYPASS	Master Filter Bypass 1 All the filter modules will be bypassed. 0 All the filters will be enabled.
17 BYPSETADDR	Bypass SetAddress in Device Mode When this bit is set, the device core uses the value in DCFG[DevAddr] bits directly for comparing the device address in the tokens. <b>NOTE:</b> This bit must be set to 1'b0.
16 U2RSTECN	If the super speed connection fails during POLL or LMP exchange, the device connects at non-SS mode. If this bit is set, then device attempts three more times to connect at SS, even if it previously failed to operate in SS mode. <b>NOTE:</b> This bit is applicable only in device mode.
15–14 FRMSCLDW	This field scales down device view of a SoF/USOF/ITP duration. For SS/HS mode- Value of 2'h3 implements interval to be 15.625 µs Value of 2'h2 implements interval to be 31.25 µs Value of 2'h1 implements interval to be 62.5 µs Value of 2'h0 implements interval to be 125 µs For FS mode, the scale-down value is multiplied by 8. 00 1024 bytes 01 512 bytes 10 256 bytes 11 128 bytes
13–12 PRTCAPDIR	Port Capability Direction 01 for Host configurations 10 for Device configurations 11 for OTG configurations For OTG, if PRTCAPDIR is 2'b11, it acts as an OTG 2.0 device with A-device or B-device determined by the IDDIG input, and host or peripheral role based on HNP. If PRTCAPDIR is 2'b01, it acts as a DRD in host mode. If PRTCAPDIR is 2'b10, it acts as a DRD in device mode. The OTG device can be programmed to enable/disable SRP and HNP by using the fields present in OCFG register. The sequence for switching modes in DRD configuration is as follows: Switching from Device to Host- 1. Reset the controller using GCTL[CORESOFTRESET].

*Table continues on the next page...*

**USBx\_GCTL field descriptions (continued)**

Field	Description
	<p>2. Set GCTL[PRTCAPDIR] to 2'b01 (Host mode).</p> <p>3. Reset the host using USBCMD[HCRESET].</p> <p>4. Follow the steps in "Initializing Host Registers".</p> <p>Switching from Host to Device-</p> <p>1. Reset the controller using GCTL[CORESOFTRESET].</p> <p>2. Set GCTL[PRTCAPDIR] to 2'b10 (Device mode).</p> <p>3. Reset the device by setting DCTL[CSFTRST].</p> <p>4. Follow the steps in "Register Initialization".</p>
11 CORESOFTRESET	<p>Core Soft Reset</p> <p>0 No soft reset</p> <p>1 Soft reset to core</p> <p>Clears the interrupts and all the CSRs except the following registers-</p> <p>GCTL</p> <p>GUCTL</p> <p>GSTS</p> <p>GGPIO</p> <p>GUID</p> <p>GUSB2PHYCFGn registers</p> <p>GUSB3PIPECTLn registers</p> <p>DCFG</p> <p>DCTL</p> <p>DEVTEN</p> <p>DSTS</p> <p>When you reset PHYs (using GUBS3PHYCFG or GUSB3PIPECTL registers), you must keep the core in reset state until PHY clocks are stable. This controls the bus, ram, and mac domain resets.</p> <p><b>NOTE:</b> This bit is for debug purpose only. Use USBCMD.HCRESET in xHCI Mode and DCTL.SoftReset in device mode for soft reset.</p>
10 SOFITPSYNC	The bit is set to 0. The core keeps the UTMI PHY on the first port in a non-suspended state whenever there is a SuperSpeed port that is not in Rx.Detect, SS.Disable and U3.
9 —	This field is reserved.
8 —	This field is reserved.
7–6 RAMCLKSEL	<p>RAM Clock Select</p> <p>00 bus clock</p> <p>01 pipe clock</p> <p>10 pipe/2 clock</p> <p>11 Reserved</p> <p>On USB-reset, hardware clears these bits to 2'b00.</p>

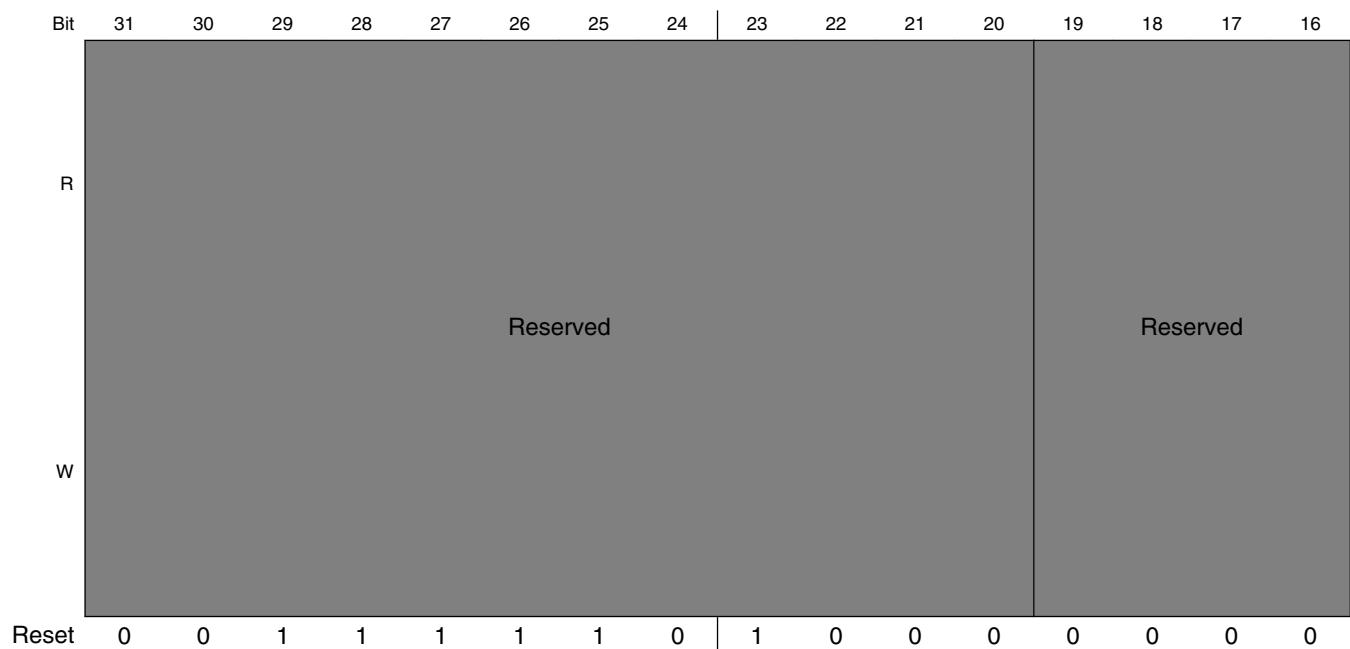
*Table continues on the next page...*

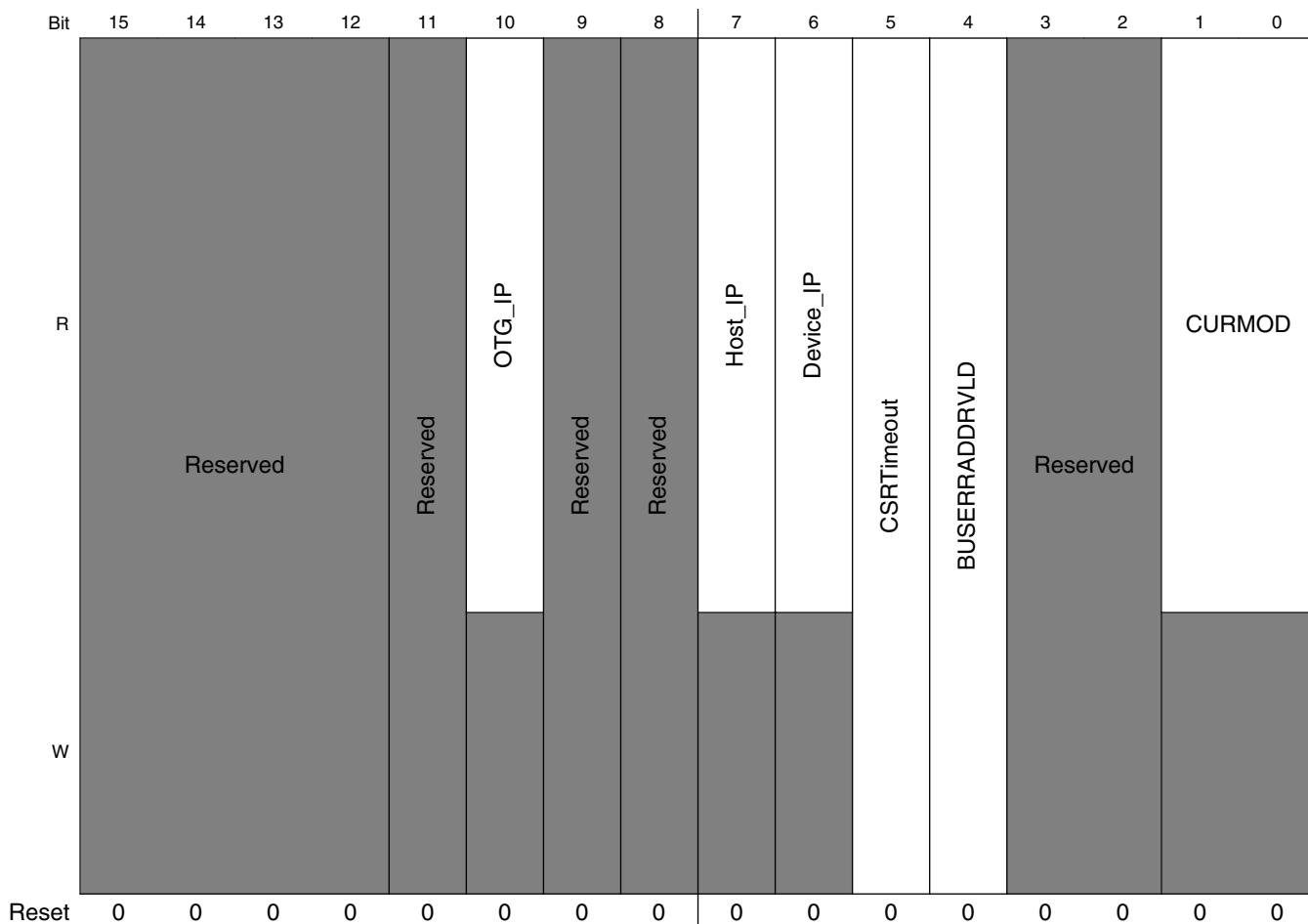
**USBx\_GCTL field descriptions (continued)**

Field	Description
	<b>NOTE:</b> In host mode, this bit must be set to 2'b00, that is, the ram_clk must be assigned to bus_clk only. The reason is if the SS port0 goes to P3, the pipe_clk is shutdown, and the USB 2.0 ports cannot operate.
5–4 —	This field is reserved.
3 DISSCRAMBLE	Disable Scrambling Transmit request to Link Partner on next transition to Recovery or Polling.
2 U2EXIT_LFPS	This bit is added to improve interoperability with a third party host controller. This host controller in U2 state while performing receiver detection generates an LFPS glitch of about 4μs duration. This causes the device to exit from U2 state because the LFPS filter value is 248ns. With the new functionality enabled, the device can stay in U2 while ignoring this glitch from the host controller.  If this bit is, 0 The link treats 248ns LFPS as a valid U2 exit. 1 The link waits for 8μs of LFPS before it detects a valid U2 exit.
1 —	This field is reserved.
0 DSBLCLKGTNG	Disable Clock Gating When this bit is set to 1 and the core is in Low Power mode, internal clock gating is disabled. Set this bit to 1'b1 after Power On Reset.

### 36.2.14 Global Status Register (USBx\_GSTS)

Address: Base address + C118h offset



**USBx\_GSTS field descriptions**

Field	Description
31–20 —	This field is reserved.
19–12 —	This field is reserved.
11 —	This field is reserved.
10 OTG_IP	OTG Interrupt Pending This field indicates that there is a pending interrupt pertaining to OTG in OEVT register.
9 —	This field is reserved.
8 —	This field is reserved.
7 Host_IP	Host Interrupt Pending This field indicates that there is a pending interrupt pertaining to xHC in the Host event queue.
6 Device_IP	Device Interrupt Pending This field indicates that there is a pending interrupt pertaining to peripheral (device) operation in the Device event queue.

Table continues on the next page...

**USBx\_GSTS field descriptions (continued)**

Field	Description
5 CSRTimeout	CSR Timeout When this bit is 1, it indicates that software performed a write or read to a core register that could not be completed within 17'h1ffff bus clock cycles.
4 BUSERRADDRVLD	Bus Error Address Valid Indicates that the GBUSERRADDR register is valid and reports the first bus address that encounters a bus error.
3–2 —	This field is reserved.
CURMOD	Current Mode of Operation 0 Device mode 1 Host mode

**36.2.15 Global User Control Register 1 (USBx\_GUCTL1)**

Address: Base address + C11Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		FILTER_SE0_FSLS_EOP	TX_IPGAP_LINECHECK_DIS	DEV_TRB_OUT_SPR_IND	DEV_FORCE_20_CLK_FOR_30_CLK	P3_IN_U2	DEV_L1_EXIT_BY_HW	IP_GAP_ADD_ON		DEV_LSP_TAIL_LOCK_DIS	NAK_PER_ENH_FS	NAK_PER_ENH_HS	Reserved		PARKMODE_DISABLE_HS
W			0	0	0	0	0	0	0	0	0	0	0	1	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved					L1_SUSP_THRLD_EN_FOR_HOST	L1_SUSP_THRLD_FOR_HOST		HC_ERRATA_ENABLE	HC_PARCHK_DISABLE	OVRLD_L1_SUSP_COM	LOA_FILTER_EN		
W								1	1	0	0	0	1	0	1	0
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0

**USBx\_GUCTL1 field descriptions**

Field	Description
31–30 —	This field is reserved.

*Table continues on the next page...*

**USBx\_GUCTL1 field descriptions (continued)**

Field	Description
29 FILTER_SE0_FSL_S_EOP	<p>0 Default behaviour, no change in Linestate check for SE0 detection in FS/LS.</p> <p>1 Feature enabled, FS/LS SE0 is filtered for 2 clocks for detecting EOP.</p> <p>This bit is applicable for FS/LS operation. If this feature is enabled, then SE0 on the linestate is validated for 2 consecutive utmi clock edges for EOP detection. This feature is applicable only in FS in device mode and FS/LS mode of operation in host mode. - Device mode: FS - If GUCTL1[FILTER_SE0_FSL_S_EOP] is set, then for device LPM handshake, the core will ignore single SE0 glitch on the linestate during transmit. Only 2 or more SE0 is considered as a valid EOP on FS. - Host mode: FS/LS - If GUCTL1[FILTER_SE0_FSL_S_EOP] is set, then the core will ignore single SE0 glitch on the linestate during transmit. Only 2 or more SE0 is considered as a valid EOP on FS/LS port. Enable this feature if the LineState has SE0 glitches during transmission. This bit is quasi-static, i.e., should not be changed during device operation.</p>
28 TX_IPGAP_LINECHECK_DIS	<p>0 Default behaviour, no change in Linestate check</p> <p>1 Feature enabled, 2.0 MAC disables Linestate check during HS transmit</p> <p>This bit is applicable for HS operation of u2mac. If this feature is enabled, then the 2.0 mac operating in HS ignores the UTMI Linestate during the transmit of a token (during token-to-token and token-to-data IPGAP). When enabled, the controller implements a fixed 40-bit TxEndDelay after the packet is given on UTMI and ignores the Linestate during this time. This feature is applicable only in HS mode of operation.</p> <p>Device mode: If GUCTL1.TX_IPGAP_LINECHECK_DIS is set, then for device LPM handshake, the core will ignore the linestate after TX and wait for a fixed clocks ( 40 bit times equivalent) after transmitting ACK on utmi.</p> <p>Host mode: If GUCTL1.TX_IPGAP_LINECHECK_DIS is set, then the ipgap between (tkn to tkn/data) is added by 40 bit times of TXENDDELAY, and linestate is ignored during this 40 bit times delay.</p> <p>Enable this bit if the LineState will not reflect the expected line state (J) during transmission. This bit is quasi-static, i.e., should not be changed during device operation.</p>
27 DEV_TRB_OUT_SPR_IND	<p>0 Default behaviour, no change in TRB status dword</p> <p>1 Feature enabled, OUT TRB status indicates Short Packet</p> <p>This bit is applicable for device mode only (and ignored in host mode). If the device application (SW/HW) wants to know if a short packet was received for an OUT in the TRB status itself, then this feature can be enabled, so that a bit is set in the TRB writeback in the buf_size dword. Bit[26] - SPR of the {trbstatus, RSVD, SPR, PCM1, bufsize} dword will be set during an OUT transfer TRB write back if this is the last TRB used for that transfer descriptor. This bit is quasi-static, i.e., should not be changed during device operation.</p>
26 DEV_FORCE_20_CLK_FOR_30_CLK	<p>0 Default behaviour, Uses 3.0 clock when operating in 2.0 mode</p> <p>1 Feature enabled</p> <p>This bit is applicable (and to be set) for device mode (DCFG.Speed != SS) only. In the 3.0 device core, if the core is programmed to operate in 2.0 only (i.e., Device Speed is programmed to 2.0 speeds in DCFG[Speed]), then setting this bit makes the internal 2.0 (utmi) clock to be routed as the 3.0 (pipe) clock. Enabling this feature allows the pipe3 clock to be not-running when forcibly operating in 2.0 device mode. Note: When using this feature, all pipe3 inputs must be in inactive mode, esp. pipe3 clocks not running and pipe3_phystatus_async must be tied to 0. This bit should not be set if the core is programmed to operate in SuperSpeed mode (even when it falls back to 2.0). This bit is quasi-static, i.e., should not be changed during operation.</p>
25 P3_IN_U2	<p>0: Default behaviour, When SuperSpeed link is in U2 , PowerState P2 is attempted on the PIPE Interface.</p> <p>1: When SuperSpeed link is in U2, PowerState P3 is attempted if GUSB3PIPECTL[17] is set.</p>

*Table continues on the next page...*

**USBx\_GUCTL1 field descriptions (continued)**

Field	Description
	Setting this bit enables P3 Power State when the SuperSpeed link is in U2. Another Power Saving option. Check with your PHY vendor before enabling this option. When setting this bit to 1 to enable P3 in P2, GUSB3PIPECTL[27] should be set to 0 to make sure that the U2 exit is attempted in P0. This bit should be set only when GCTL.SOFITPSYNC=1 or GFLADJ.GFLADJ_REFCLK_LPM_SEL=1.
24 DEV_L1_EXIT_BY_HW	<p>0 Default behaviour, disables device L1 hardware exit logic 1 feature enabled</p> <p>This bit is applicable for device mode (2.0) only. This field enables device controller sending remote wakeup for L1 if the device becomes ready for sending/accepting data when in L1 state. If the host expects the device to send remote wkp signalling to resume after going into L1 in flow controlled state, then this bit can be set to send the remote wake signal automatically when the device controller becomes ready. This HW remote wake feature is applicable only to bulk and interrupt transfers, and not for Isoch/Control.</p> <p>When control transfers are in progress, the LPM will be rejected (NYET response). Only after control transfers are completed (either with ACK/STALL), LPM will be accepted.</p> <p>For Isoch transfers, the host needs to do the wake-up and start the transfer. Device controller will not do remote-wakeup when Isoch endpoints get ready. The device SW needs to keep the GUSB2PHYCFG[EnbISlpM] reset in order to keep the PHY clock to be running for keeping track of SOF intervals.</p> <p>This bit is quasi-static, i.e., should not be changed during device operation.</p>
23–21 IP_GAP_ADD_ON	This register field is used to add on to the default inter packet gap setting in the USB 2.0 MAC.
20 DEV_LSP_TAIL_LOCK_DIS	<p>0 Default behaviour, enables device lsp lock logic for tail TRB update 1 Fix disabled</p> <p>This is a bug fix for STAR 9000716195 that affects the CSP mode for OUT endpoints in device mode. The issue is that tail TRB index is not synchronized with the cache Scratchpad bytecount update. If the fast-forward request comes in-between the bytecount update on a newly fetched TRB and the tail-index write update in TPF, the RDP works on an incorrect tail index and misses the byte count decrement for the newly fetched TRB in the fast-forwarding process. This fix needs to be present all the times.</p>
19 NAK_PER_ENH_FS	<p>When this bit is set to:</p> <p>1 Enables performance enhancement for FS async endpoints in the presence of NAKs 0 Enhancement not applied</p> <p>If a periodic endpoint is present, and if a bulk endpoint which is also active is being NAKed by the device, then this could result in a decrease in performance of other Full Speed bulk endpoint which is ACKed by the device. Setting this bit to 1, will enable the host controller to schedule more transactions to the async endpoints (bulk/ control) and hence will improve the performance of the bulk endpoint. This control bit should be enabled only if the existing performance with the default setting is not sufficient for your FullSpeed application. Setting this bit will only control, and is only required for Full Speed transfers.</p>
18 NAK_PER_ENH_HS	<p>When this bit is set to:</p> <p>1 Enables performance enhancement for HS async endpoints in the presence of NAKs 0 Enhancement not applied</p> <p>If a periodic endpoint is present, and if a bulk endpoint which is also active is being NAKed by the device, then this could result in a decrease in performance of other High Speed bulk endpoint which is ACKed by the device. Setting this bit to 1, will enable the host controller to schedule more transactions to the async endpoints (bulk/ control) and hence will improve the performance of the bulk endpoint. This control bit should be enabled only if the existing performance with the default setting is not sufficient for your HighSpeed application. Setting this bit will only control, and is only required for High Speed transfers.</p>

Table continues on the next page...

**USBx\_GUCTL1 field descriptions (continued)**

Field	Description
17 —	This field is reserved.
16 PARKMODE_DISABLE_HS	<p>This bit is used only in host mode. When this bit is set to '1' all HS bus instances park mode are disabled.</p> <p>To improve performance in park mode, the xHCI scheduler queues in three requests of 4 packets each for High Speed asynchronous endpoints in a micro-frame. But if a device is slow and it NAKs more than 3 times, then it is rescheduled only in the next micro-frame. This could decrease the performance of a slow device even further.</p> <p>In a few high speed devices (such as Sandisk Cruzer Blade 4GB VID:1921, PID:21863 and Flex Drive VID:3744, PID:8552) when an IN request is sent within 900ns of the ACK of the previous packet, these devices send a NAK. When connected to these devices, if required, the software can disable the park mode if you see performance drop in your system. When park mode is disabled, pipelining of multiple packet is disabled and instead one packet at a time is requested by the scheduler. This allows up to 12 NAKs in a micro-frame and improves performance of these slow devices.</p>
15 —	This field is reserved.
14–9 —	This field is reserved.
8 L1_SUSP_THRLD_EN_FOR_HOST	<p>This bit is used only in host mode. The host controller asserts the utmi_l1_suspend_n and utmi_sleep_n output signals to the PHY in the L1 state.</p> <p>0 Disable 1 Enable</p>
7–4 L1_SUSP_THRLD_FOR_HOST	<p>This field is effective only when the L1_SUSP_THRLD_EN_FOR_HOST bit is set to 1.</p> <p>1111 State is normal working 0110 State is L2 Suspend 0101 State is L1 Suspend 1011 State is L1 Sleep</p>
3 HC_ERRATA_ENABLE	<p>Host ELD Enable When this bit is set to 1, it enables the Exit Latency Delta (ELD) support defined in the xHCI 1.0 Errata. This bit is used only in the host mode.</p>
2 HC_PARCHK_DISABLE	<p>Host Parameter Check Disable When this bit is set to '0' (by default), the xHC checks that the input slot/EP context fields comply to the xHCI Specification. Upon detection of a parameter error during command execution, the xHC generates an event TRB with completion code indicating 'PARAMETER ERROR'. When the bit is set to '1', the xHC does not perform parameter checks and does not generate 'PARAMETER ERROR' completion code.</p>
1 OVRLD_L1_SUSP_COM	<p>If this bit is set, the utmi_l1_suspend_com_n is overloaded with the utmi_sleep_n signal. This bit is usually set if the PHY stops the port clock during L1 sleep condition.</p> <p><b>NOTE:</b> The recommended connection for the SUSPENDM/SLEEPM signals to the PHY with respect to this bit is as follows. For Non-0 Ports, connect utmi_sleep_n[n] to SLEEPM[n]</p>

*Table continues on the next page...*

**USBx\_GUCTL1 field descriptions (continued)**

Field	Description
	<p>(utmi_suspend_n[n] &amp; utmi_l1_suspend_n[n]) to SUSPENDM[n]</p> <p>USB2 PHYCLK[n] to utmi_clk[n] GUCTL1[OVRLD_L1_SUSP_COM] impacts only Port0. For Port0:</p> <p>For PHY, GUSB2PHYCFGn[U2_FREECLK_EXISTS]=1:</p> <p>With this connection, the PHY keeps PLL active so that FREECLK is always available irrespective of suspend/sleep.</p> <p>Connect USB2 PHY COMMONONN to 0.</p> <p>Connect utmi_sleep_n[0] to SLEEPM[0].</p> <p>Connect (utmi_suspend_n[0] &amp; utmi_l1_suspend_n[0]) to SUSPENDM[0].</p> <p>Connect USB2 PHY FREECLK to utmi_clk[0].</p> <p>Leave utmi_suspend_com_n, utmi_l1_suspend_com_n unconnected.</p> <p>GUCTL1[OVRLD_L1_SUSP_COM] can be set to any value.</p> <p>For Third Party PHY, GUSB2PHYCFGn[U2_FREECLK_EXISTS]=0:</p> <p>With this connection the PHY can shut off all the clocks when the required conditions are met (for example GUSB2PHYCFGn[8,6], GUCTL1[1], GFLADJ[23], GCTL[10], Suspend condition, HW LPM enable and so on).</p> <p>Connect ~utmi_suspend_com_n to SUSPENDM[0] (or equivalent).</p> <p>Connect ~utmi_l1_suspend_com_n to SLEEPM[0] (or equivalent).</p> <p>Connect PHYCLK0 (first port clock) to utmi_clk[0].</p> <p>Leave utmi_suspend_n[0], utmi_l1_suspend_n[0], utmi_sleep_n[0] unconnected.</p> <p>Set GUCTL1[OVRLD_L1_SUSP_COM] to 1'b1.</p>
0 LOA_FILTER_EN	<p>If this bit is set, the USB 2.0 port babble is checked at least three consecutive times before the port is disabled. This prevents false triggering of the babble condition when using low quality cables.</p> <p><b>NOTE:</b> This bit is valid only in host mode.</p>

**36.2.16 Global User ID Register (USBx\_GUID)**

The register is read-only register and contains user ID.

Address: Base address + C128h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 1 0 0

**USBx\_GUID field descriptions**

Field	Description
UserID	User ID

### 36.2.17 Global User Control Register (USBx\_GUCTL)

This register provides a few options for the software to control the core behavior in the Host mode. Most of the options are used to improve host inter-operability with different devices.

Address: Base address + C12Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	REFCLKPER												NoExtrDI	Reserved		SprsCrtTransEn		
W																		
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	CMdevAddr	USBHstlnAutoRetry En	EnOverlapChk	ExtCapSuptEN	InsltExtrFSBODI	DTCT		DTFT										
W																		
Reset	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0		

#### USBx\_GUCTL field descriptions

Field	Description
31–22 REFCLKPER	<p>This field indicates in terms of nano seconds the period of ref_clk.</p> <p>The default value of this register is set to 'h8 (8ns/125 MHz). This field needs to be updated during power-on initialization, if GCTL[SOFITPSYNC] or GFLADJ[GFLADJ_REFCLK_LPM_SEL] is set to '1'. The programmable maximum value is 62ns, and the minimum value is 8ns.</p> <p>Use the reference clock with a period that is an integer multiple, so that ITP can meet the jitter margin of 32ns. The allowable ref_clk frequencies whose period is not integer multiples are 16/17/19.2/24/39.7 MHz.</p> <p>This field should not be set to '0' at any time. If you never plan to use this feature, then set this field to 'h8, the default value.</p>
21 NoExtrDI	<p>No Extra Delay Between SOF and the First Packet</p> <p>Some HS devices misbehave when the host sends a packet immediately after a SOF. However, adding an extra delay between a SOF and the first packet can reduce the USB data rate and performance.</p> <p>This bit is used to control whether the host should wait for 2 microseconds before it sends the first packet after a SOF, or not. User can set this bit to one to improve the performance if those problematic devices are not a concern in the user's host environment.</p>

Table continues on the next page...

**USBx\_GUCTL field descriptions (continued)**

Field	Description
	0 Host waits for 2 microseconds after a SOF before it sends the first USB packet. 1 Host doesn't wait after a SOF before it sends the first USB packet.
20–18 —	This field is reserved.
17 SprsCtrlTransEn	Sparse Control Transaction Enable  Some devices are slow in responding to Control transfers. Scheduling multiple transactions in one microframe/frame can cause these devices to misbehave.  If this bit is set to 1'b1, the host controller schedules transactions for a Control transfer in different microframes/frames.
16 ResBwHSEPS	Reserving 85% Bandwidth for HS Periodic EPs  By default, HC reserves 80% of the bandwidth for periodic EPs. If this bit is set, the bandwidth is relaxed to 85% to accommodate two high speed, high bandwidth ISOC EPs.  USB 2.0 required 80% bandwidth allocated for ISOC traffic. If two High-bandwidth ISOC devices (HD Webcams) are connected, and if each requires 1024-bytes X 3 packets per Micro-Frame, then the bandwidth required is around 82%. If this bit is set, then it is possible to connect two Webcams of 1024bytes X 3 payload per Micro-Frame each. Otherwise, you may have to reduce the resolution of the Webcams.  This bit is valid in Host and DRD configuration and is used in host mode operation only. Ignore this bit in device mode.
15 CMdevAddr	Compliance Mode for Device Address  When this bit is 1'b1, Slot ID may have different value than Device Address if max_slot_enabled < 128.  1 Increment Device Address on each Address Device command.  0 Device Address is equal to Slot ID.  The xHCI compliance requires this bit to be set to '1'. The '0' mode is for debug purpose only. This allows you to easily identify a device connected to a port in the Lecroy or Eliisys trace during hardware debug.  This bit is valid in Host and DRD configuration and is used in host mode operation only. Ignore this bit in device mode.
14 USBHstInAutoRetryEn	Host IN Auto Retry  When set, this field enables the Auto Retry feature. For IN transfers (non-isochronous) that encounter data packets with CRC errors or internal overrun scenarios, the auto retry feature causes the Host core to reply to the device with a non-terminating retry ACK (that is, an ACK transaction packet with Retry = 1 and NumP != 0).  If the Auto Retry feature is disabled (default), the core will respond with a terminating retry ACK (that is, an ACK transaction packet with Retry = 1 and NumP = 0).  0 Auto Retry Disabled 1 Auto Retry Enabled  <b>NOTE:</b> This bit is also applicable to the device mode.
13 EnOverlapChk	Enable Check for LFPS Overlap During Remote Ux Exit  If this bit is set to,

*Table continues on the next page...*

**USBx\_GUCTL field descriptions (continued)**

Field	Description
	<p>1 The SuperSpeed link when exiting U1/U2/U3 waits for either the remote link LFPS or TS1/TS2 training symbols before it confirms that the LFPS handshake is complete. This is done to handle the case where the LFPS glitch causes the link to start exiting from the low power state. Looking for the LFPS overlap makes sure that the link partner also sees the LFPS.</p> <p>0 When the link exists U1/U2/U3 because of a remote exit, it does not look for an LFPS overlap.</p>
12 ExtCapSuptEN	<p>External Extended Capability Support Enable</p> <p>When set, this field enables extended capabilities to be implemented outside the core. A read to the first DWORD of the last internal extended capability (the "xHCI Supported Protocol Capability for USB 3.0 when the Debug Capability is not enabled, or the "Debug Capability" when it is enabled) returns a value of 4 in the Next Capability Pointer field. This indicates to software that there is another capability four DWORDs after this capability (for example, at address N+16 where N is the address of this DWORD). If enabled, an external address decoder that snoops the xHC slave interface needs to be implemented. If it sees an access to N+16 or greater, the slave access is re-routed to a piece of hardware which returns the external capability pointer register of the new capability and also handles reads/writes to this new capability and the side effects.</p> <p>If disabled, a read to the first DWORD of the last internal extended capability will return 0 in the 'Next Capability Pointer field. This indicates there are no more capabilities.</p>
11 InsltExtrFSBODI	<p>Insert Extra Delay Between FS Bulk OUT Transactions</p> <p>Some FS devices are slow to receive Bulk OUT data and can get stuck when there are consecutive Bulk OUT transactions with short inter-transaction delays. This bit is used to control whether the host inserts extra delay between consecutive Bulk OUT transactions to a FS Endpoint.</p> <p>0- Host doesn't insert extra delay between consecutive Bulk OUT transactions to a FS Endpoint.</p> <p>1- Host inserts about 12us extra delay between consecutive Bulk OUT transactions to a FS Endpoint to work around the device issue.</p> <p><b>NOTE:</b> Setting this bit to one will reduce the Bulk OUT transfer performance for most of the FS devices.</p>
10–9 DTCT	<p>Device Timeout Coarse Tuning</p> <p>This field is a Host mode parameter which determines how long the host waits for a response from device before considering a timeout. The core first checks the DTCT value. If it is 0, then the timeout value is defined by the DTFT. If it is non-zero, then it uses the following timeout values-</p> <ul style="list-style-type: none"> <li>00 0 <math>\mu</math>sec -&gt; use DTFT value instead</li> <li>01 500 <math>\mu</math>sec</li> <li>10 1.5 <math>\mu</math>sec</li> <li>11 6.5 <math>\mu</math>sec</li> </ul>
DTFT	<p>Device Timeout Fine Tuning</p> <p>This field is a Host mode parameter which determines how long the host waits for a response from device before considering a timeout. For DTFT field to take effect, DTCT must be set to 2'b00.</p> <p>The DTFT value is the number of 125 MHz clocks * 256 to count before considering a device timeout.</p> <p>For the 125 MHz clock (8 ns period), this is calculated as follows- (DTFT value) * 256 * (8 ns)</p> <p>Quick Reference-</p> <p>if DTFT = 0x2, <math>2 \times 256 \times 8 = 4\mu</math>sec timeout</p>

*Table continues on the next page...*

## USBx GUCTL field descriptions (continued)

Field	Description
	if DTFT = 0x5, $5 \times 256 \times 8 = 10\mu\text{sec}$ timeout if DTFT = 0xA, $10 \times 256 \times 8 = 20\mu\text{sec}$ timeout if DTFT = 0x10, $16 \times 256 \times 8 = 32\mu\text{sec}$ timeout if DTFT = 0x19, $25 \times 256 \times 8 = 51\mu\text{sec}$ timeout if DTFT = 0x31, $49 \times 256 \times 8 = 100\mu\text{sec}$ timeout if DTFT = 0x62, $98 \times 256 \times 8 = 200\mu\text{sec}$ timeout

### **36.2.18 Global SoC Bus Error Address Register low (USBx\_GBUSERRADDRLO)**

When the AXI Master Bus returns "Error" response, the "SoC Bus Error" is generated. In the Host mode, the host\_system\_err port indicates this condition. In addition, it is also indicated in the USBSTS[HSE] field. In the Device mode, the GSTS[BusErrAddrVld] field is the only indication of the SoC Bus Error.

Due to the nature of AXI, it is possible that multiple AXI transactions are active at a time. The USB 3.0 controller does not keep track of the start address of all outstanding transactions. Instead, it keeps track of the start address of the DMA transfer associated with all active transactions. It is this address that is reported in the GBUSERRADDR when a bus error occurs.

For example, if the USB 3.0 controller initiates a DMA transfer to write 1k of packet data starting at buffer address 0xABCD0000, and this DMA is broken up into multiple 256B bursts on the AXI, then if a bus error occurs on any of these associated AXI transfers, the GBUSERRADDR reflects the DMA start address of 0xABCD0000 regardless of which AXI transaction received the error.

Address: Base address + C130h offset

## USBx GBUSERRADDRLO field descriptions

Field	Description
BUSERRADDR	<p>Bus Address - Low</p> <p>This 64-bit register contains the lower 32 bits of the first bus address that encountered a SoC bus error. It is valid when the GSTS[BusErrAddrVld] field is 1.</p> <p>It can only be cleared by resetting the core.</p>

### 36.2.19 Global SoC Bus Error Address Register high (USBx\_GBUSERRADDRHI)

It represents the remaining bits of global SoC bus error address register.

Address: Base address + C134h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
	BUSERRADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### USBx\_GBUSERRADDRHI field descriptions

Field	Description
BUSERRADDR	Bus Address - High  This 64-bit register contains the higher 32 bits of the first bus address that encountered a SoC bus error. It is valid when the GSTS[BusErrAddrVld] field is 1. It can only be cleared by resetting the core.

### 36.2.20 Global SS Port to Bus Instance Mapping Register - Low (USBx\_GPRTBIMAPLO)

This is an alternate register for the GPRTBIMAP register.

#### NOTE

For reset values, refer to the corresponding values in the GPRTBIMAP register.

Address: Base address + C138h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	Reserved																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### USBx\_GPRTBIMAPLO field descriptions

Field	Description
31–4	This field is reserved.
BINUM1	SS USB instance number for Port. Value set as 0.

### 36.2.21 Global SS Port to Bus Instance Mapping Register - High (USBx\_GPRTBIMAPHI)

This is an alternate register for the GPRTBIMAP register.

#### NOTE

For reset values, refer to the corresponding values in the GPRTBIMAP register.

Address: Base address + C13Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_GPRTBIMAPHI field descriptions

Field	Description
31–4 —	This field is reserved.
BINUM9	SS USB Instance Number for Port 9.

### 36.2.22 Global Hardware Parameters Register 0 (USBx\_GHWPARAMS0)

Address: Base address + C140h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
	DWC_USB3_AWIDTH										DWC_USB3_SDWIDTH					
W																
Reset	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									DWC_USB3_SBUS_TYPE		DWC_USB3_MBUS_TYPE		DWC_USB3_MODE			
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0

#### USBx\_GHWPARAMS0 field descriptions

Field	Description
31–24 DWC_USB3_AWIDTH	Master/Slave address bus width: 64 bit

Table continues on the next page...

**USBx\_GHWPARAMS0 field descriptions (continued)**

Field	Description
23–16 DWC_USB3_ SDWIDTH	Slave bus (Register access bus) data bus width: 32 bit
15–8 DWC_USB3_ MDWIDTH	Master bus (DMA bus) data bus width  It selects the data bus width of the master bus interface. 33-bit option is used only for Hub configuration.  The possible values are: <ul style="list-style-type: none"> <li>• 32 32-bits</li> <li>• 33 33-bits</li> <li>• 64 64-bits</li> <li>• 128 128-bits</li> </ul>
7–6 DWC_USB3_ SBUS_TYPE	Slave bus (Register access bus) interface type  It selects the chip slave bus interface type. The slave bus is used for register programming.  The settings not shown are reserved.  00 AHB
5–3 DWC_USB3_ MBUS_TYPE	Master bus (DMA bus) interface type  It selects the chip master bus interface type. The master bus is used for DMA.  The settings not shown are reserved.  01 AXI
DWC_USB3_ MODE	Mode of operation  It selects the controller mode for USB 3.0.  <b>NOTE:</b> It is configurable based on license(s) purchased.  The settings not shown are reserved.  10 DRD

### 36.2.23 Global Hardware Parameters Register 1 (USBx\_GHWPARAMS1)

This register contains the hardware configuration options

Address: Base address + C144h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DWC_USB3_EN_DBC	DWC_USB3_RM_OPT_FEATURES	Reserved	DWC_USB3_RAM_BUS_CLKS_SYNC	DWC_USB3_MAC_RAM_CLKS_SYNC	DWC_USB3_MAC_PHY_CLKS_SYNC	DWC_USB3_EN_PWROPT	DWC_USB3_SDRAM_TYP	DWC_USB3_NUM_RAMS	DWC_USB3_DEVICE_NUM_INT						
W																
Reset	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DWC _USB3 _DEVI _CE_ NUM _INT	DWC_USB3_ ASPACEWIDTH	DWC_USB3_ REQINFOWIDTH	DWC_USB3_ DATAINFOWIDTH	DWC_USB3_ BURSTWIDTH	DWC_USB3_ IDWIDTH1										
W																

Reset    1    1    0    0    1    0    0    1    0    0    1    1    1    0    1    1

### USBx\_GHWPARAMS1 field descriptions

Field	Description
31 DWC_USB3_EN_ DBC	Enables xHCI debug capability 0 No 1 Yes
30 DWC_USB3_RM_ OPT_FEATURES	It specifies whether to remove optional features. 0 No 1 Yes  When this parameter is enabled, the User ID register, General Purpose Input/Output ports, and SOF toggle and counter ports are removed.
29 —	This field is reserved.
28 DWC_USB3_RAM_ BUS_CLKS_SYNC	It specifies whether the RAM clock and the Bus clock are synchronous to each other. 0 No 1 Yes
27 DWC_USB3_MAC_ RAM_CLKS_SYNC	It specifies whether the MAC clock and the RAM clock are synchronous to each other. 0 No 1 Yes

Table continues on the next page...

**USBx\_GHWPARAMS1 field descriptions (continued)**

Field	Description
26 DWC_USB3_MAC_PHY_CLKS_SYNC	It specifies whether the MAC clock and the PHY clock are synchronous to each other. 0 No 1 Yes
25–24 DWC_USB3_EN_PWROPT	Power optimization mode It specifies the power optimization mode. If clock gating only is selected, RAM and PHY clocks are gated when the core is inactive during U1,U2, or U3 states. The possible values are as follows: 0 No power optimization 1 Clock gating only
23 DWC_USB3_SPRAM_TYP	Synchronous static RAM type It selects the FIFO synchronous static RAM type. The possible values are as follows: 00 2-Port RAM (2Port-RAM) 01 Single-port RAM (SPRAM) 10-11 Reserved
22–21 DWC_USB3_NUM_RAMs	Number of RAMs It selects the number of RAMs. The possible values are 1, 2 and 3.
20–15 DWC_USB3_DEVICE_NUM_INT	Number of device mode event buffers It selects the number of event buffers in device mode. The possible values are 1, 2,..., 32.
14–12 DWC_USB3_ASPACEWIDTH	It selects the address space port width of the master and slave bus interfaces. The possible values are 1, 2, 3, 4, 5 and 6.
11–9 DWC_USB3_REQINFOWIDTH	It selects the Request/Response info port width of the master and slave bus interfaces. The possible values are 4, 5 and 6.
8–6 DWC_USB3_DATAINFOWIDTH	It selects the data info port width of the master and slave bus interfaces. The possible values are 1, 2, 3, 4, 5 and 6.
5–3 DWC_USB3_BURSTWIDTH	DWC_USB3_BURSTWIDTH 1 It selects the burst port width of the master and slave bus interfaces. The possible values are 1,2, 3, 4, 5, 6, 7 and 8.
DWC_USB3_IDWIDTH1	Master ID port width It selects the ID port width of the master bus interface. This parameter limits the number of pipelined AXI transfers. The GSBUSCFG1[PipeTransLimit] field can only be programmed to a value less than or equal to two to the power of DWC_USB3_IDWIDTH. The possible values are 4, 5, 6, 7 and 8.

### 36.2.24 Global Hardware Parameters Register 2 (USBx\_GHWPARAMS2)

Address: Base address + C148h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R																DWC_USB3_USERID																		
W																																		
Reset	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	1	0	1	0			

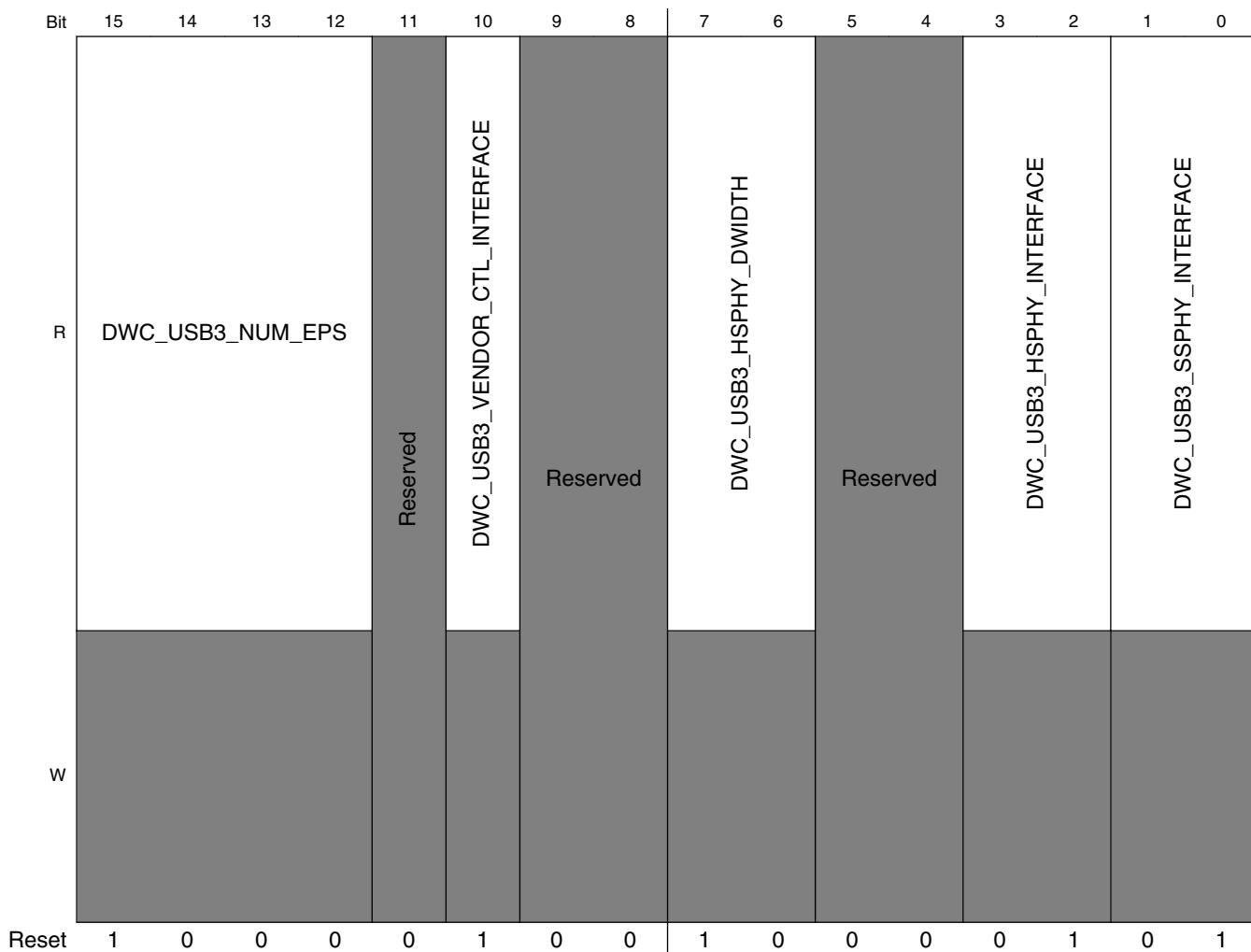
#### USBx\_GHWPARAMS2 field descriptions

Field	Description
DWC_USB3_USERID	Global user ID (GUID) register's power-on initialization value It specifies the global user ID (GUID) register's power-on initialization value. The value is set as 32'h130290a.

### 36.2.25 Global Hardware Parameters Register 3 (USBx\_GHWPARAMS3)

Address: Base address + C14Ch offset

Bit	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	Reserved	DWC_USB3_CACHE_TOTAL_XFER_RESOURCES								DWC_USB3_NUM_IN_EPS				DWC_USB3_NUM_EPS			
W																	
Reset	0	0	0	0	0	1	0	0		0	0	0	1	0	0	0	0



### USBx\_GHWPARAMS3 field descriptions

Field	Description
31 —	This field is reserved.
30–23 DWC_USB3_CASCADE_TOTAL_XFER_RESOURCES	It selects the maximum number of transfer resources in the core. The value is set as 8.
22–18 DWC_USB3_NUM_IN_EPS	Number of device mode active IN endpoints It specifies the maximum number of device mode IN endpoints active at any time, including control endpoint 0, which is always present. The value is set as 4.
17–12 DWC_USB3_NUM_EPS	Number of device mode endpoints It specifies the number of device mode single directional endpoints, including OUT and IN endpoint. The value is set as 8.

Table continues on the next page...

**USBx\_GHWPARAMS3 field descriptions (continued)**

Field	Description
11 —	This field is reserved.
10 DWC_USB3_ VENDOR_CTL_ INTERFACE	The bit enables the UTMI+ PHY vendor control interface. The value is enabled and value is set as 1.
9–8 —	This field is reserved.
7–6 DWC_USB3_ HSPHY_ DWIDTH	It specifies the data width of the UTMI+ PHY interface. 10 8/16-bits All other settings are reserved.
5–4 —	This field is reserved.
3–2 DWC_USB3_ HSPHY_ INTERFACE	It specifies the High-Speed PHY interface(s). The value is set as 1 for UTMI+.
DWC_USB3_ SSPHY_ INTERFACE	It specifies the superSpeed PHY interface. The value is set as 1 for PIPE3. In USB 2.0 only mode, set to 0 else select the PIPE3 interface.

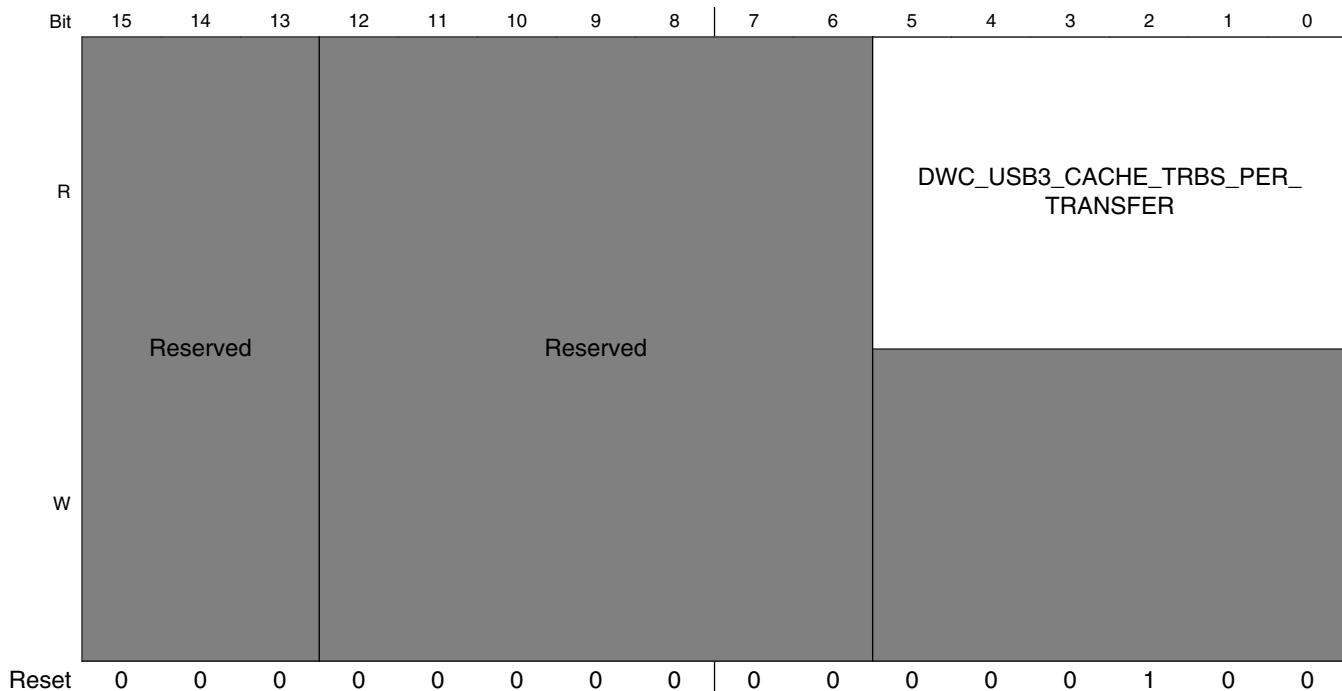
### 36.2.26 Global Hardware Parameters Register 4 (USBx\_GHWPARAMS4)

The register is read-only.

Address: Base address + C150h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DWC_USB3_BMU_LSP_DEPTH			DWC_USB3_BMU_PTL_DEPTH			DWC_USB3_EN_ISOC_SUPT		Reserved	Reserved	DWC_USB3_NUM_SS_USB_INSTANCES					Reserved
W																
Reset	0	1	0	0	0	1	1	1	1	0	0	0	0	0	1	0

## USB Memory Map/Register Definition



### USBx\_GHWPARAMS4 field descriptions

Field	Description
31–28 DWC_USB3_BMU_LSP_DEPTH	It specifies the depth of the BMU-LSP status buffer. The value is set as 4.
27–24 DWC_USB3_BMU_PTL_DEPTH	It specifies the depth of the BMU-PTL source/sink buffers. The value is set as 8.
23 DWC_USB3_EN_ISOC_SUPT	It enables isochronous endpoint capability. The capability is enabled by default, set as 1.
22 —	This field is reserved.
21 —	This field is reserved.
20–17 DWC_USB3_NUM_SS_USB_INSTANCES	Number of SuperSpeed USB bus instances It specifies the number of SuperSpeed USB bus instances. The value is set as 1.
16–13 —	This field is reserved.
12–6 —	This field is reserved.
DWC_USB3_CACHE_TRBS_PER_TRANSFER	Number of cached TRBs per transfer It selects the number of transfer request blocks (TRBs) per transfer that can be cached within the core. The values is set as 4.

### 36.2.27 Global Hardware Parameters Register 5 (USBx\_GHWPARAMS5)

The register is read-only.

Address: Base address + C154h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										DWC_USB3_DFQ_FIFO_DEPTH					
W																
Reset	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DWC_USB3_TXQ_FIFO_DEPTH					DWC_USB3_RXQ_FIFO_DEPTH					DWC_USB3_BMU_BUSGM_DEPTH					
W																
Reset	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0

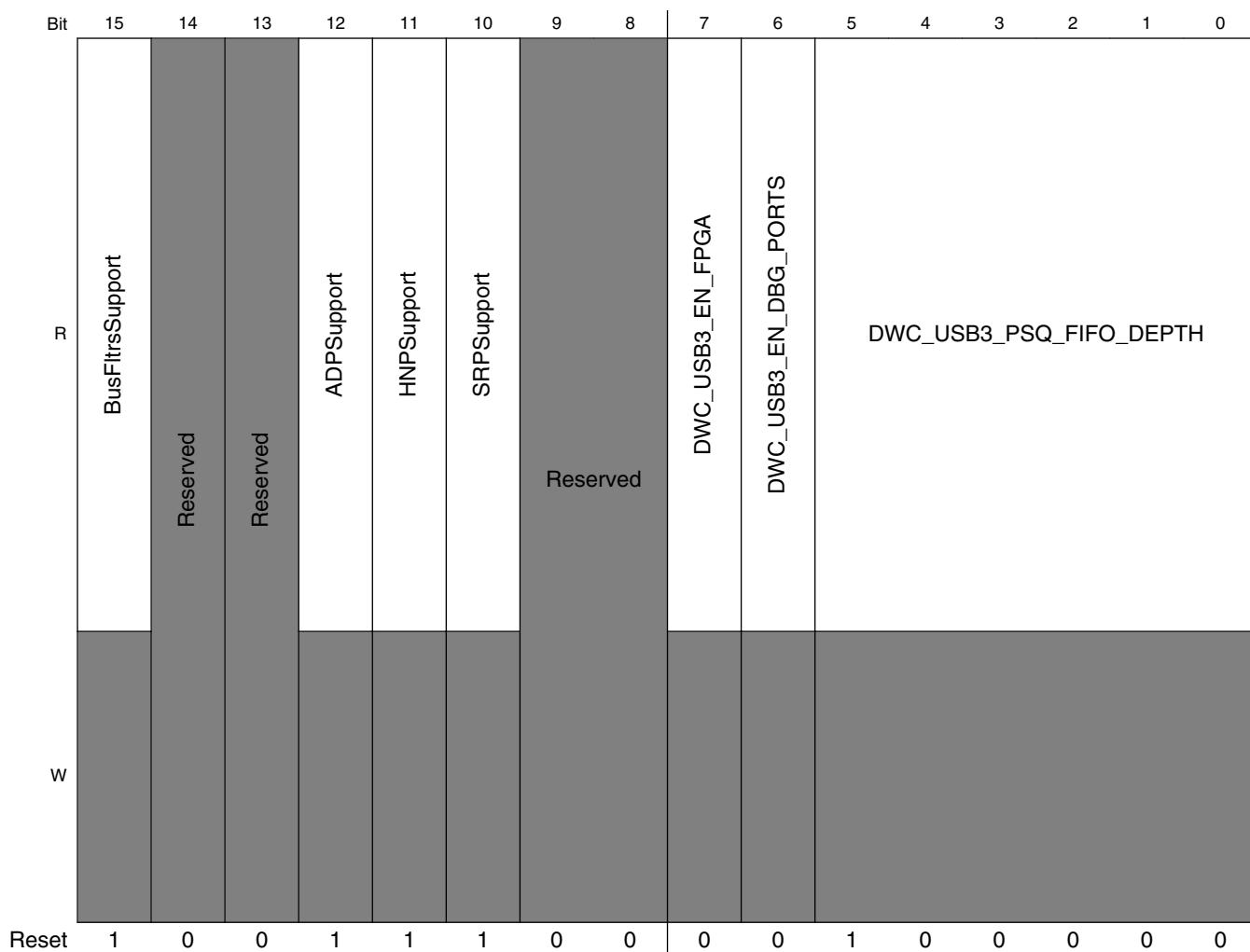
#### USBx\_GHWPARAMS5 field descriptions

Field	Description
31–28 —	This field is reserved.
27–22 DWC_USB3_DFQ_FIFO_DEPTH	It specifies the size of the BMU descriptor fetch request queue. The specified depth is allocated in the data FIFO RAM and defines the number of descriptor fetch commands the scheduler can queue to the BMU. The value is set as 16.
21–16 DWC_USB3_DWQ_FIFO_DEPTH	It specifies the size of the BMU descriptor write queue. The specified depth is allocated in the data FIFO RAM (32-/64-/128-bits wide) and defines the number of descriptor write commands the scheduler can queue to the BMU. value is set as 32.
15–10 DWC_USB3_TXQ_FIFO_DEPTH	It specifies the size of the BMU Tx request queue. The specified depth is allocated in the data FIFO RAM (32-/64-/128-bits wide) and defines the number of Tx commands the scheduler can queue to the BMU. The value is set as 16.
9–4 DWC_USB3_RXQ_FIFO_DEPTH	It specifies the size of the BMU Rx request queue. The specified depth is allocated in the data FIFO RAM (32-/64-/128-bits wide) and defines the number of Rx commands the scheduler can queue to the BMU. The value is set as 16.
DWC_USB3_BMU_BUSGM_DEPTH	It specifies the depth of the BMU-BUSGM source/sink buffers. The FIFOs are 32-/64-/128-bits wide, matching the bus master data width. The value is set as 8.

### 36.2.28 Global Hardware Parameters Register 6 (USBx\_GHWPARAMS6)

Address: Base address + C158h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R									DWC_USB3_RAM0_DEPTH							
W																
Reset	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0



### USBx\_GHWPARAMS6 field descriptions

Field	Description
31–16 DWC_USB3_ RAM0_DEPTH	Total RAM0 depth, set as 2308. It specifies the depth of RAM0. In Device, Host, and DRD configuration RAM0 contains: <ul style="list-style-type: none"><li>• 3-RAM configuration: Descriptor cache</li><li>• 2-RAM configuration: Descriptor cache and RxFIFOs</li><li>• 1-RAM configuration: Descriptor cache, TxFIFOs, and RxFIFOs</li></ul>
15 BusFltrsSupport	It specifies whether to add a filter for VBUS and ID related control inputs from the PHY. <ul style="list-style-type: none"><li>• UTMI+ PHY: This signal is from the PHY.</li></ul> 0 No 1 Yes
14 —	This field is reserved.
13 —	This field is reserved.

Table continues on the next page...

**USBx\_GHWPARAMS6 field descriptions (continued)**

Field	Description
12 ADPSupport	It enables internal ADP capability of the USB 3.0 core. When it is enabled, the core incorporates ADP controller logic and provides ADP control signals.
11 HNPSupport	HNP Support Enabled The application uses this bit to determine the USB 3.0 core's HNP support. 0 HNP support is not enabled 1 HNP support is enabled
10 SRPSupport	SRP Support Enabled The application uses this bit to determine the USB 3.0 core's SRP support. 0 SRP support is not enabled 1 SRP support is enabled
9–8 —	This field is reserved.
7 DWC_USB3_EN_FPGA	Hardware validation/driver development with an FPGA platform 0 No 1 Yes
6 DWC_USB3_EN_DBG_PORTS	It is used for FPGA hardware validation of the core. 0 No 1 Yes
DWC_USB3_PSQ_FIFO_DEPTH	It specifies the size of the BMU protocol status queue. The value is set as 32. The specified depth is allocated in the data FIFO RAM 32-/64-/128-bits wide) and defines the number of header and status dwords the PTL can queue to the LSP.

### 36.2.29 Global Hardware Parameters Register 7 (USBx\_GHWPARAMS7)

Address: Base address + C15Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
	DWC_USB3_RAM2_DEPTH															DWC_USB3_RAM1_DEPTH																
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	1	0	1

**USBx\_GHWPARAMS7 field descriptions**

Field	Description
31–16 DWC_USB3_RAM2_DEPTH	Total RAM2 depth. It specifies the depth of RAM2, set as 776.
DWC_USB3_RAM1_DEPTH	Total RAM1 depth. It specifies the depth of RAM1, set as 1101.

### 36.2.30 Global High-Speed Port to Bus Instance Mapping Register - Low (USBx\_GPRTBIMAP\_HSLO)

This is an alternate register for the GPRTBIMAP\_HS register.

#### NOTE

For reset values, refer to the corresponding values in the GPRTBIMAP\_HS register.

Address: Base address + C180h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_GPRTBIMAP\_HSLO field descriptions

Field	Description
31–4 —	This field is reserved.
BINUM1	HS USB Instance Number for Port 1, value set as 0.

### 36.2.31 Global High-Speed Port to Bus Instance Mapping Register - High (USBx\_GPRTBIMAP\_HSHI)

This is an alternate register for the GPRTBIMAP\_HS register.

Address: Base address + C184h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_GPRTBIMAP\_HSHI field descriptions

Field	Description
31–4 —	This field is reserved.
BINUM9	HS USB Instance Number for Port 9.

### 36.2.32 Global USB2 PHY Configuration Register (USBx\_GUSB2PHYCFG)

The SoC must program this register before starting any transaction.

Address: Base address + C200h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PHYSOFTRST	U2_FREECLK_EXISTS														
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								ENBLSLPM	0	0	0	0	0			
W									SUSPENDUSB20	0	0	0	0	0		
Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0

## USBx\_GUSB2PHYCFG field descriptions

Field	Description
31 PHYSOFTRST	UTMI PHY Soft Reset. It causes the usb2phy_reset signal to be asserted to reset a UTMI PHY.
30 U2_FREECLK_EXISTS	Specifies USB 2.0 PHY free-running PHY clock exists, set as 1.
29–25 -	This field is reserved.
24–22 LSTRD	<p>LS Turnaround Time. This field indicates the value of the Rx-to-Tx packet gap for LS devices. The encoding is as follows:</p> <ul style="list-style-type: none"> <li>0 2 bit times</li> <li>1 2.5 bit times</li> <li>2 3 bit times</li> <li>3 3.5 bit times</li> <li>4 4 bit times</li> <li>5 4.5 bit times</li> <li>6 5 bit times</li> <li>7 5.5 bit times</li> </ul> <p><b>NOTE:</b> This field is applicable only in Host mode. For normal operation, to work with most LS devices the default value is set as 0 (2 bit times).</p> <p>The programmable LS device inter-packet gap and turnaround delays are provided to support some legacy LS devices that might require different delays than the default/fixed ones. For instance, the Open LS mouse requires 3 bit times of inter-packet gap to work correctly. Include your PHY delays when programming the LSIPD/LSTRDTIM values. For example, if your PHY's TxEndDelay in LS mode is 30 UTMI CLKs, then subtract this delay (~1 LS bit time) from the device's delay requirement.</p>
21–19 LSIPD	<p>LS Inter-Packet Time.mThis field indicates the value of Tx-to-Tx packet gap for LS devices. The encoding is as follows:</p> <ul style="list-style-type: none"> <li>0 2 bit times</li> <li>1 2.5 bit times</li> <li>2 3 bit times</li> <li>3 3.5 bit times</li> <li>4 4 bit times</li> <li>5 4.5 bit times</li> <li>6 5 bit times</li> <li>7 5.5 bit times</li> </ul> <p><b>NOTE:</b> This field is applicable only in Host mode. For normal operation to work with most LS devices the field is set to 2 (3 bit times).</p> <p>The programmable LS device inter-packet gap and turnaround delays are provided to support some legacy LS devices that might require different delays than the default/fixed ones. For instance, the AOpen LS mouse requires 3 bit times of inter-packet gap to work correctly. Include your PHY delays when programming the LSIPD/LSTRDTIM values. For example, if your PHY's TxEndDelay in LS mode is 30 UTMI CLKs, then subtract this delay (~1 LS bit time) from the device's delay requirement.</p>

*Table continues on the next page...*

**USBx\_GUSB2PHYCFG field descriptions (continued)**

Field	Description
18–9 -	This field is reserved.
8 ENBLSLPM	Enable utmi_sleep_n and utmi_l1_suspend_n. The application uses this bit to control utmi_sleep_n and utmi_l1_suspend_n assertion to the PHY in the L1 state. 0: utmi_sleep_n and utmi_l1_suspend_n assertion from the core is not transferred to the external PHY. 1: utmi_sleep_n and utmi_l1_suspend_n assertion from the core is transferred to the external PHY.
7 Reserved	This read-only field is reserved and always has the value 0.
6 SUSPENDUSB20	Suspend USB2.0 HS/FS/LS PHY When set, USB2.0 PHY enters Suspend mode if Suspend conditions are valid. If it is set to '1', then the application should clear this bit after power-on reset. Application needs to set it to '1' after the core initialization is completed. <b>NOTE:</b> In host mode, on reset, this bit is set to '1'. Software can override this bit after reset.
5–4 Reserved	This read-only field is reserved and always has the value 0.
3 PHYIF	PHY Interface If UTMI+ is selected, the application uses this bit to configure the core to support a UTMI+ PHY with an 8- or 16-bit interface. 0 8 bits 1 16 bits
-	This field is reserved.

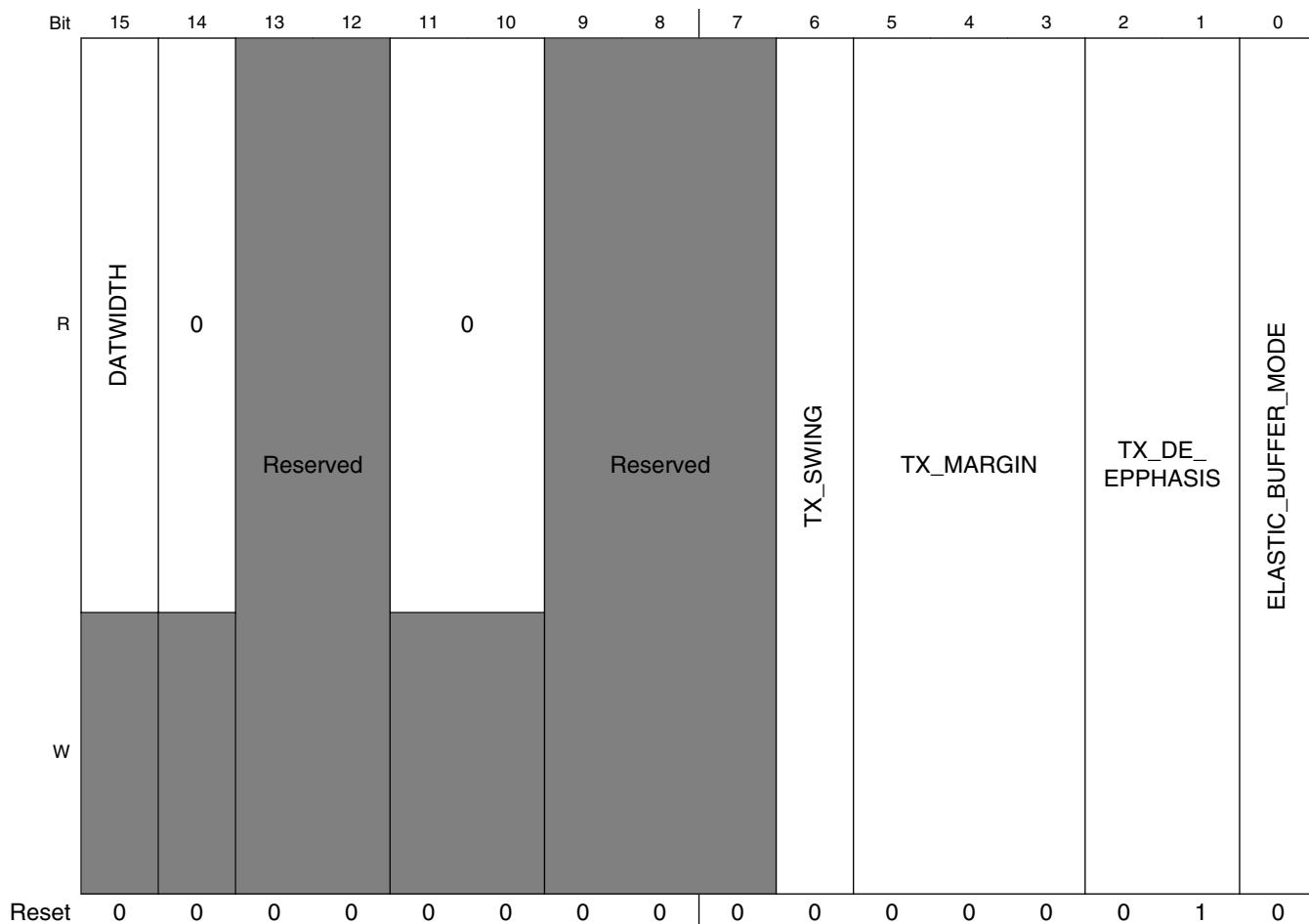
### 36.2.33 Global USB 3.0 PIPE Control Register (USBx\_GUSB3PIPECTL)

The application uses this register to configure the USB3 PHY and PIPE interface.

Address: Base address + C2C0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PHYSOFTRST	Reserved	U2SSInactP3ok	DisRxDetP3	0	-	u1u2exitfail_to_recov	-	-	-	-	-	0	-	DATWIDTH	-
W	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0

Reset    0    0    0    0    0    0    0    1    0    0    0    0    1    1    0    0

**USBx\_GUSB3PIPECTL field descriptions**

Field	Description
31 PHYSOFRST	USB3 PHY Soft Reset. After setting this bit to '1', the software needs to clear this bit.
30 —	This field is reserved.
29 U2SSInactP3ok	P3 OK for U2/SS.Inactive 0 During link state U2/SS.Inactive, put PHY in P2 (Default) 1 During link state U2/SS.Inactive, put PHY in P3
28 DisRxDetP3	Disabled receiver detection in P3 0 If PHY is in P3 and Core needs to perform receiver detection, Core will perform receiver detection in P3. (Default) 1 If PHY is in P3 and Core needs to perform receiver detection, Core will change PHY power state to P2 and then perform receiver detection. After receiver detection, Core will change PHY power state to P3.
27–26 Reserved	This read-only field is reserved and always has the value 0.
25 u1u2exitfail_to_recov	U1U2exitfail to Recovery

Table continues on the next page...

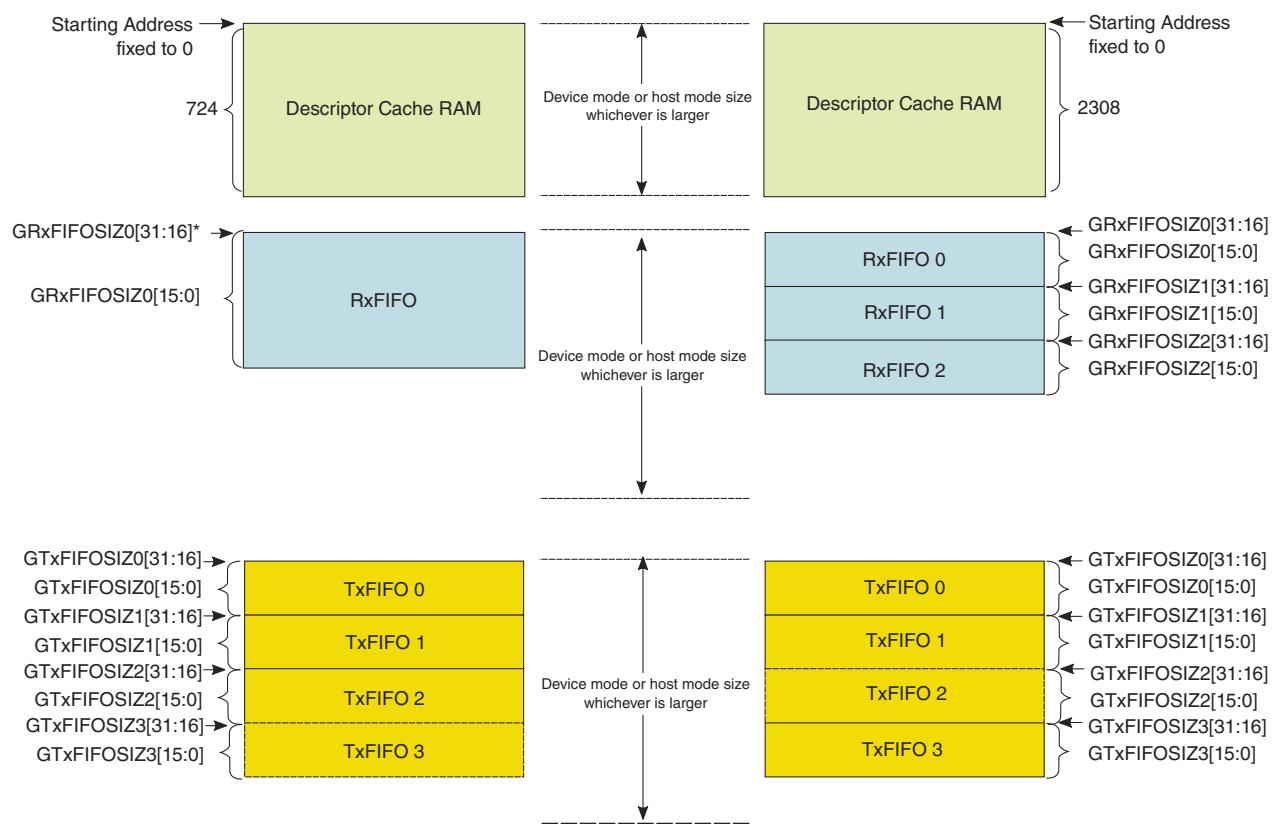
**USBx\_GUSB3PIPECTL field descriptions (continued)**

Field	Description
	When set, and U1/U2 LFPS handshake fails, the LTSSM transitions from U1/U2 to Recovery instead of SS Inactive. If Recovery fails, then the LTSSM can enter SS.Inactive. This is an enhancement only. It prevents interoperability issue if the remote link does not do proper handshake.
24 -	This bit should be always set to 1.
23–20 -	
19–18 -	This bit is read-only and should be always set to 1.
17 Reserved	This read-only field is reserved and always has the value 0.
16–15 DATWIDTH	PIPE Data Width 2'b00: 32 bits 2'b01: 16 bits 2'b10: 8 bits  One clock after reset, these bits receive the value.
14 Reserved	This bit should always read 0.  This read-only field is reserved and always has the value 0.
13–12 -	This field is reserved.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–7 —	This field is reserved.
6 TX_SWING	Tx Swing Refer to the PIPE3 specification.
5–3 TX_MARGIN	Tx Margin[2:0] Refer to Table 5-3 of the PIPE3 specification.
2–1 TX_DE_EPPHASIS	Tx Deemphasis The value driven to the PHY is controlled by the LTSSM during USB3 Compliance mode. (Refer to Table 5-3 of the PIPE3 specification.)
0 ELASTIC_BUFFER_MODE	Elastic Buffer Mode

**36.2.34 Global Transmit FIFO Size Register (USBx\_GTXFIFOSIZn)**

This register specifies the RAM start address and depth

## **USB Memory Map/Register Definition**



**Figure 36-3. GTxFIFOSIZn and GRxFIFOSIZn Usage in DRD Mode Without Debug Capability**

Address: Base address + C300h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXFSTADDR_N															TXFDEP_N																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0		

## USBx\_GTXFIFOSIZn field descriptions

Field	Description
31–16 TXFSTADDR_N	<p>Transmit FIFO RAM Start Address</p> <p>This field contains the memory start address for TxFIFO in MDWIDTH-bit words.</p>
TXFDEP_N	<p>TxFIFO Depth</p> <p>This field contains the depth of TxFIFO in MDWIDTH-bit words.</p> <p>Minimum value: 32</p> <p>Maximum value: 32,768</p>

### 36.2.35 Global Receive FIFO Size Register (USBx\_GRXFIFOSIZn)

This register specifies the RAM start address and depth.

Address: Base address + C380h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFSTADDR_N																RXFDEP_N															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	1	

#### USBx\_GRXFIFOSIZn field descriptions

Field	Description
31–16 RXFSTADDR_N	RxFIFOOn RAM Start Address  This field contains the memory start address for RxFIFOOn in MDWIDTH-bit words.
RXFDEP_N	RxFIFO Depth  This fields contains the depth of RxFIFOOn in MDWIDTH-bit words.  Minimum value: 32  Maximum value: 16,384

### 36.2.36 Global Event Buffer Address (Low) Register (USBx\_GEVNTADRLO)

This register holds the Event Buffer DMA Address pointer. Software must initialize this address once during power-on initialization. Software must not change the value of this register after it is initialized. Software must only use the GEVNTCOUNT register for event processing.

Address: Base address + C400h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVNTADRLO																EVNTADRHI															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

#### USBx\_GEVNTADRLO field descriptions

Field	Description
EVNTADRLO	Event Buffer Address  Holds the lower 32 bits of start address of the external memory for the Event Buffer. During operation, hardware does not update this address.

### 36.2.37 Global Event Buffer Address (High) Register (USBx\_GEVNTADRHI)

This register holds the Event Buffer DMA Address pointer. Software must initialize this address once during power-on initialization. Software must not change the value of this register after it is initialized. Software must only use the GEVNTCOUNT register for event processing.

Address: Base address + C404h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_GEVNTADRHI field descriptions

Field	Description
EVNTADRHI	Event Buffer Address Holds the higher 32 bits of start address of the external memory for the Event Buffer. During operation, hardware does not update this address.

### 36.2.38 Global Event Buffer Size Register (USBx\_GEVNTSIZ)

This register holds the Event Buffer Size and the Event Interrupt Mask bit. During power-on initialization, software must initialize the size with the number of bytes allocated for the Event Buffer. The Event Interrupt Mask will mask the interrupt, but events are still queued. After configuration, software must preserve the Event Buffer Size value when changing the Event Interrupt Mask.

Address: Base address + C408h offset

Bit	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R	EVNTINTRPTMASK	SK															
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R																	
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

#### USBx\_GEVNTSIZ field descriptions

Field	Description
31 EVNTINTRPTMASK	Event Interrupt Mask When set to '1', this prevents the interrupt from being generated. However, even when the mask is set, the events are queued.
30–16 —	This field is reserved.
EVENTSIZ	Event Buffer Size in bytes Holds the size of the Event Buffer in bytes; must be a multiple of four. This is programmed by software once during initialization. The minimum size of the event buffer is 32 bytes.

### **36.2.39 Global Event Buffer Count Register (USBx\_GEVNTCOUNT)**

This register holds the number of valid bytes in the Event Buffer. During initialization, software must initialize the count by writing 0 to the Event Count field. Each time the hardware writes a new event to the Event Buffer, it increments this count. Most events are four bytes, but some events may span over multiple four byte entries. Whenever the count is greater than zero, the hardware raises the corresponding interrupt line (depending on the EvntIntMask bit in the GEVNTSIZ register). On an interrupt, software processes one or more events out of the Event Buffer. Afterwards, software must write the Event Count field with the number of bytes it processed.

Clock crossing delays may result in the interrupt's continual assertion after software acknowledges the last event. Therefore, when the interrupt line is asserted, software must read the GEVNTCOUNT register and only process events if the GEVNTCOUNT is greater than 0.

Address: Base address + C40Ch offset

## USBx GEVNTCOUNT field descriptions

Field	Description
31–16 —	This field is reserved.
EVNTCOUNT	<p>Event Count</p> <p>When read, returns the number of valid events in the Event Buffer (in bytes). When written, hardware decrements the count by the value written.</p> <p>The interrupt line remains high when count is not 0.</p>

### **36.2.40 Global Hardware Parameters Register 8 (USBx\_GHWPARAMS8)**

Address: Base address + C600h offset

### USBx\_GHWPARAMS8 field descriptions

Field	Description
DWC_USB3_DCACHE_DEPTH_INFO	The read-only value defines the minimum RAM0 requirement. The value of the field is set as 2308.

### 36.2.41 Global Device TX FIFO DMA Priority Register (USBx\_GTXFIFOPRIDEV)

This register specifies the relative DMA priority level among the Device TXFIFOs (one per IN endpoint). Each register bit[n] controls the priority (1: high, 0: low) of each TXFIFO[n]. When multiple TXFIFOs compete for DMA service at a given time (that is, multiple TXQs contain TX DMA requests and their corresponding TXFIFOs have space available), the TX DMA arbiter grants access on a packet-basis in the following manner:

High-priority TXFIFOs are granted access using round-robin arbitration

Low-priority TXFIFOs are granted access using round-robin arbitration only after the high-priority TXFIFOs have no further processing to do (that is, either the TXQs are empty or the corresponding TXFIFOs are full).

For scatter-gather packets, the arbiter grants successive DMA requests to the same FIFO until the entire packet is completed.

When configuring periodic IN endpoints, software must set register bit[n]=1, where n is the TXFIFO assignment. This ensures that the DMA for isochronous or interrupt IN endpoints are prioritized over bulk or control IN endpoints. The register size corresponds to the number of Device IN endpoints.

#### NOTE

Since the device mode uses only one RXFIFO, there is no Device RXFIFO DMA Priority Register.

Address: Base address + C610h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															GTXFIFOPRIDEV
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBx\_GTXFIFOPRIDEV field descriptions**

Field	Description
31–4 —	This field is reserved.
GTXFIFOPRIDEV	Device TXFIFO priority 0 Low (Default) 1 High

### 36.2.42 Global Host TX FIFO DMA Priority Register (USBx\_GTXFIFOPRIHST)

This register specifies the relative DMA priority level among the Host TXFIFOs (one per USB bus instance) within the associated speed group (SS or HS/FS/LS). Each register bit[n] controls the priority (1: high, 0: low) of TXFIFO[n] within a speed group. When multiple TXFIFOs compete for DMA service at a given time (that is, multiple TXQs contain TX DMA requests and their corresponding TXFIFOs have space available), the TX DMA arbiter grants access on a packet-basis in the following manner:

1. Among the FIFOs in the same speed group (SS or HS/FS/LS):
  - a. High-priority TXFIFOs are granted access using round-robin arbitration
  - b. Low-priority TXFIFOs are granted access using round-robin arbitration only after the high- priority TXFIFOs have no further processing to do (that is, either the TXQs are empty or the corresponding TXFIFOs are full).
2. The TX DMA arbiter prioritizes the SS speed group or HS/FS/LS speed group according to the ratio programmed in the GDMAHLRATIO register.

For scatter-gather packets, the arbiter grants successive DMA requests to the same FIFO until the entire packet is completed. The register size corresponds to the number of configured USB bus instances; for example, in the default configuration, there are 3 USB bus instances (1 SS, 1 HS, and 1 FS/LS).

Address: Base address + C618h offset

Bit	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R W	Reserved																
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### USBx\_GTXFIFOPRIHST field descriptions

Field	Description
31–3 —	This field is reserved.
GTXFIFOPRIHST	Host TXFIFO priority 1 high priority 0 low priority)

### 36.2.43 Global Host RX FIFO DMA Priority Register (USBx\_GRXFIFOPRIHST)

This register specifies the relative DMA priority level among the Host RXFIFOs (one per USB bus instance) within the associated speed group (SS or HS/FS/LS). Each register bit[n] controls the priority (1: high, 0: low) of RXFIFO[n] within a speed group. When multiple RXFIFOs compete for DMA service at a given time (that is, multiple RXQs contain RX DMA requests and their corresponding RXFIFOs have data available), the RX DMA arbiter grants access on a packet-basis in the following manner:

1. Among the FIFOs in the same speed group (SS or HS/FS/LS):
  - a. High-priority RXFIFOs are granted access using round-robin arbitration
  - b. Low-priority RXFIFOs are granted access using round-robin arbitration only after high-priority RXFIFOs have no further processing to do (that is, either the RXQs are empty or the corresponding RXFIFOs do not have the required data).
2. The RX DMA arbiter prioritizes the SS speed group or HS/FS/LS speed group according to the ratio programmed in the GDMAHLRATIO register.

For scatter-gather packets, the arbiter grants successive DMA requests to the same FIFO until the entire packet is completed. This register is present only when the core is configured to operate in the host mode (includes DRD and OTG modes). The register size corresponds to the number of configured USB bus instances; for example, in the default configuration, there are 3 USB bus instances (1 SS, 1 HS, and 1 FS/LS).

Address: Base address + C61Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															GRXFIFOPRIHST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USBx GRXFIFOPRIHST field descriptions

Field	Description
31–3 —	This field is reserved.
GRXFIFOPRIHST	Host RXFIFO priority 1 high priority 0 low priority)

### **36.2.44 Global Host FIFO DMA High-Low Priority Ratio Register (USBx\_GDMAHLRATIO)**

This register specifies the relative priority of the SS FIFOs with respect to the HS/FS/LS FIFOs. The DMA arbiter prioritizes the HS/FS/LS round-robin arbiter group every DMA High-Low Priority Ratio grants as indicated in the register separately for TX and RX.

To illustrate, consider that all FIFOs are requesting access simultaneously, and the ratio is 4. SS gets priority for 4 packets, HS/FS/LS gets priority for 1 packet, SS gets priority for 4 packets, HS/FS/LS gets priority for 1 packet, and so on.

If FIFOs from both speed groups are not requesting access simultaneously then,

- If SS got grants 4 out of the last 4 times, then HS/FS/LS get the priority on any future request.
  - If HS/FS/LS got the grant last time, SS gets the priority on the next request.
  - If there is a valid request on either SS or HS/FS/LS, a grant is always awarded; there is no idle. This register is present if the core is configured to operate in host mode.

Address: Base address + C624h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R																																		
W	Reserved																HSTRXFIFO				Reserved		HSTTXFIFO											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0			

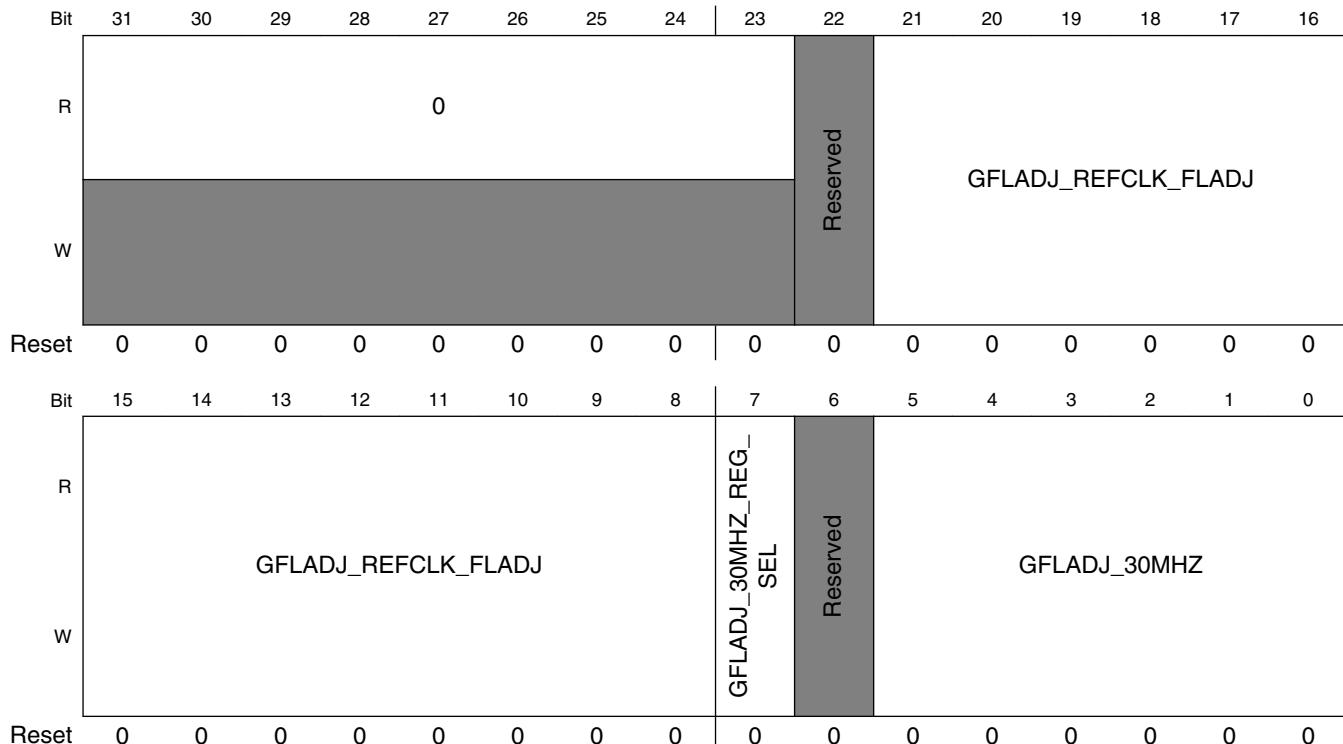
## USBx\_GDMAHLRATIO field descriptions

Field	Description
31–13 —	This field is reserved.
12–8 HSTRXFIFO	Host RXFIFO DMA High-Low priority ratio
7–5 —	This field is reserved.
HSTTXFIFO	Host TXFIFO DMA High-Low priority ratio

### 36.2.45 Global Frame Length Adjustment Register (USBx\_GFLADJ)

It provides an option to override the fladj\_30mhz\_reg sideband signal.

Address: Base address + C630h offset



#### USBx\_GFLADJ field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 —	This field is reserved.
21–8 GFLADJ_ REFCLK_FLADJ	This field indicates the frame length adjustment to be applied when SOF/ITP counter is running on the ref_clk.
7 GFLADJ_ 30MHZ_REG_ SEL	This field selects whether to use a hard-coded value of 20h (32 decimal) or the value in GFLADJ[GFLADJ_30MHZ] to adjust the frame length for the SOF/ITP. 0 The controller uses the hard coded value 20h (32 decimal). which gives a SOF cycle time of 60000. 1 The controller uses the value in GFLADJ[GFLADJ_30MHZ]
6 —	This field is reserved.
GFLADJ_30MHZ	This field indicates the value that is used for frame length adjustment when GFLADJ_30MHZ_REG_SEL = 1. Each step of this field's value corresponds to 16 high-speed bit times. The SOF cycle time (the number

Table continues on the next page...

**USBx\_GFLADJ field descriptions (continued)**

Field	Description
	<p>of SOF counter clock periods to generate a SOF microframe length) is equal to 59488 + the value in this field. When GFLADJ_30MHZ_REG_SEL = 0, a hard-coded value of 20h (32 decimal) is used, which gives a SOF cycle time of 60000. For details on how to set this value, refer to section 5.2.4, "Frame Length Adjustment Register (FLADJ)," of the xHCI Specification.</p> <p>000000 Frame length is 59488 HS bit times      000001 Frame length is 59504 HS bit times      000010 Frame length is 59520 HS bit times      ...      011111 Frame length is 59984 HS bit times      100000 Frame length is 60000 HS bit times      ...      111110 Frame length is 60480 HS bit times      111111 Frame length is 60496 HS bit times</p>

**36.2.46 Device Configuration Register (USBx\_DCFG)**

This register configures the core in Device mode after power-on or after certain control commands or enumeration. Do not make changes to this register after initial programming.

Address: Base address + C700h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								IGMSTRMPP	LPMCAP	NUMP				INTR NUM	
W	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INTRNUM				Reserved		DEVADDR					DEVSPD				
W	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

## USBx\_DCFG field descriptions

Field	Description
31–24 —	This field is reserved.
23 IGMSTRMPP	<p>Ignore Stream PP</p> <p>This bit only affects stream-capable bulk endpoints.</p> <p>When this bit is set to '0' and the controller receives a Data Packet with the Packet Pending (PP) bit set to 0 for OUT endpoints, or it receives an ACK with the NumP field set to 0 and PP set to 0 for IN endpoints, the core attempts to search for another stream (CStream) to initiate to the host. However, there are two situations where this behavior is not optimal:</p> <ul style="list-style-type: none"> <li>• When the host is setting PP=0 even though it has not finished the stream, or</li> <li>• When the endpoint on the device is configured with one transfer resource and therefore does not have any other streams to initiate to the host.</li> </ul> <p>When this bit is set to '1', the core ignores the Packet Pending bit for the purposes of stream selection and does not search for another stream when it receives DP(PP=0) or ACK(NumP=0, PP=0). This can enhance the performance when the device system bus bandwidth is low or the host responds to the core's ERDY transmission very quickly.</p>
22 LPMCAP	<p>LPM Capable</p> <p>The application uses this bit to control the USB 3.0 core LPM capabilities. If the core operates as a non-LPM-capable device, it cannot respond to LPM transactions.</p> <p>0: LPM capability is not enabled. 1: LPM capability is enabled.</p>
21–17 NUMP	<p>Number of Receive Buffers</p> <p>This bit indicates the number of receive buffers to be reported in the ACK TP.</p> <p>The USB 3.0 controller uses this field if GRXTHRCFG[USBRXPKTCNTSEL] is set to '0'. The application can program this value based on RxFIFO size, buffer sizes programmed in descriptors, and system latency.</p> <p>For an OUT endpoint, this field controls the number of receive buffers reported in the NumP field of the ACK TP transmitted by the core. Note: This bit is used in host mode when Debug Capability is enabled.</p>
16–12 INTRNUM	<p>Interrupt number</p> <p>indicates interrupt/EventQ number on which non-endpoint-specific device-related interrupts (see DEVT) are generated.</p>
11–10 —	This field is reserved.
9–3 DEVADDR	<p>Device Address</p> <p>The application must perform the following:</p> <ul style="list-style-type: none"> <li>• Program this field after every SetAddress request.</li> <li>• Reset this field to zero after USB reset.</li> </ul>
DEVSPD	<p>Device Speed</p> <p>Indicates the speed at which the application requires the core to connect, or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected.</p> <p>100: SuperSpeed (USB 3.0 PHY clock is 125 MHz or 250 MHz) 000: High-speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) 001: Full-speed (USB 2.0 PHY clock is 30 MHz or 60 MHz)</p>

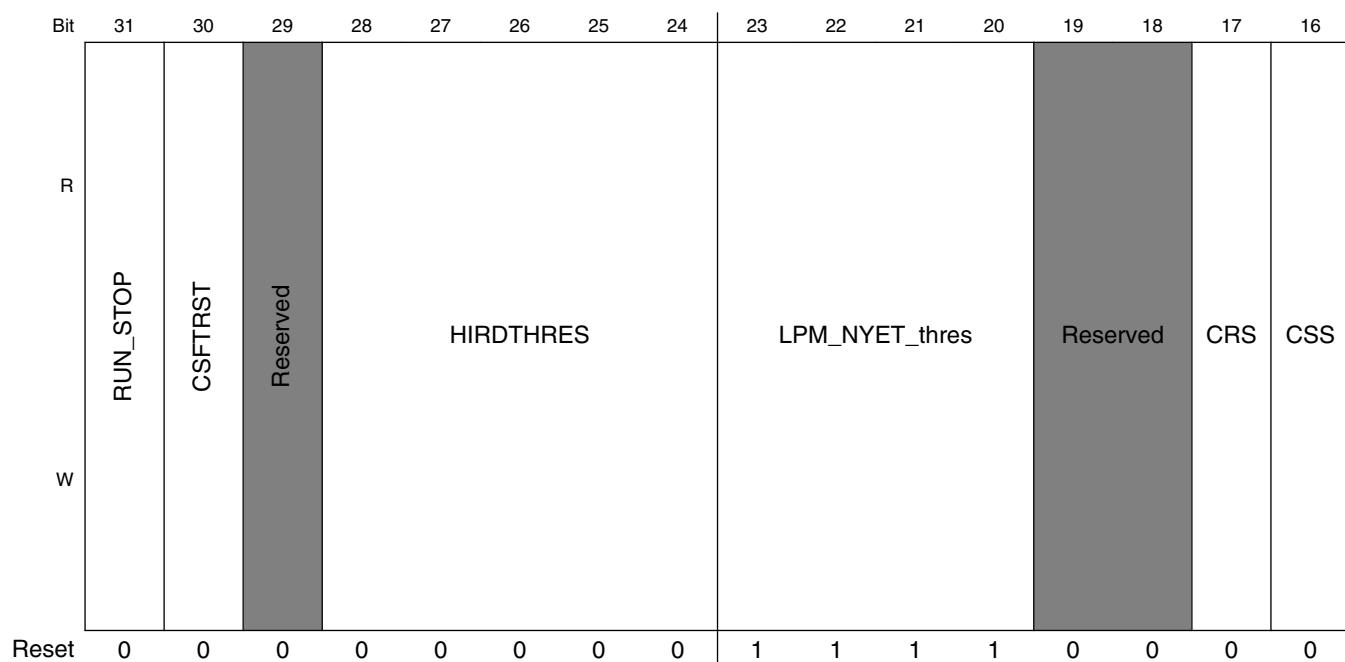
### 36.2.47 Device Control Register (USBx\_DCTL)

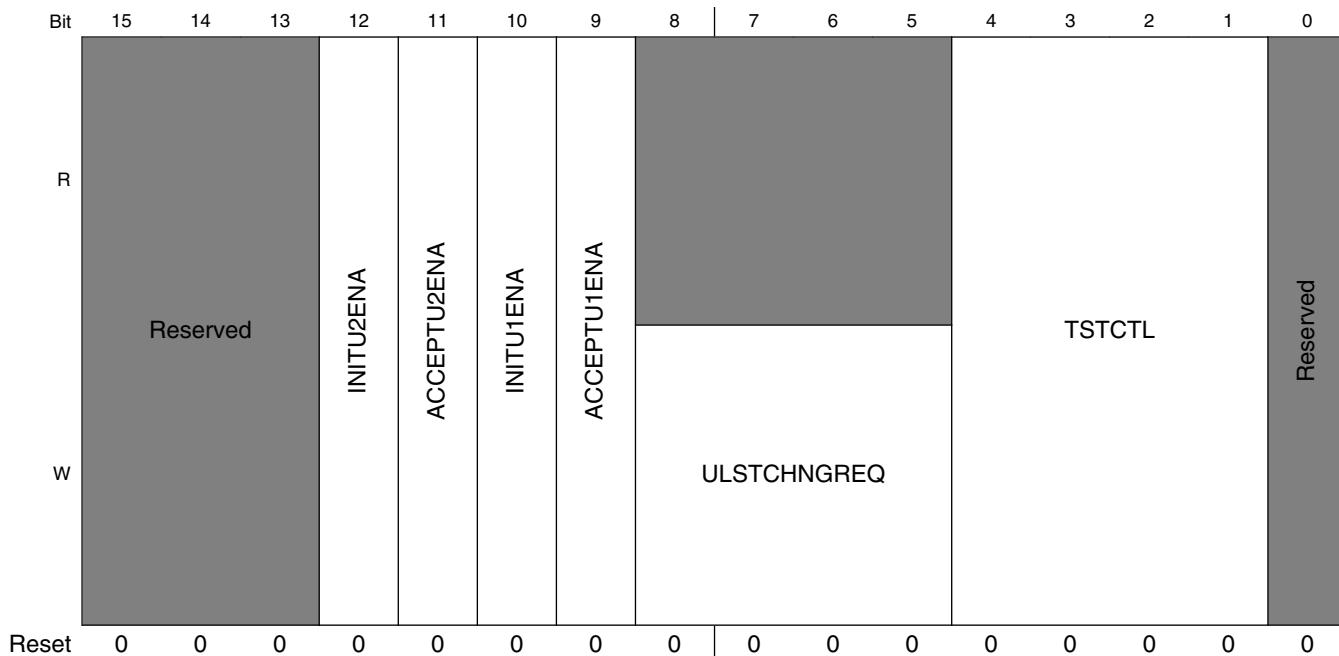
The table below lists the minimum duration under various conditions for which the soft disconnect (SftDiscon) bit must be set for the USB host to detect a device disconnect. To accommodate clock jitter, it is recommended that the application add extra delay to the specified minimum duration.

**Table 36-6. Minimum Duration for Soft Disconnect**

Operating Speed	Device State	Minimum Duration
Super speed	Suspended	30 ms
Super speed	Idle	30 ms
Super speed	Transmit or Receive	30 ms
High speed	Suspended	10 ms
High speed	Idle	10 ms
High speed	Not Idle or Suspended (Performing transactions)	10 ms
Full-speed/Low-speed	Suspended	10 ms
Full-speed/Low-speed	Idle	10 ms
Full-speed/Low-speed	Not Idle or Suspended (Performing transactions)	10 ms

Address: Base address + C704h offset





### USBx\_DCTL field descriptions

Field	Description
31 RUN_STOP	<p>The software writes 1 to this bit to start the device controller operation.</p> <p>To stop the device controller operation, the software must remove any active transfers and write 0 to this bit. When the controller is stopped, it sets the DSTS[DevCtrlHlt] bit when the core is idle and the lower layer finishes the disconnect process.</p> <p>The RUN_STOP bit must be used in following cases as specified:</p> <ol style="list-style-type: none"> <li>After power-on reset and CSR initialization, the software must write 1 to this bit to start the device controller. The controller does not signal connect to the host until this bit is set.</li> <li>The software uses this bit to control the device controller to perform a soft disconnect. When the software writes 0 to this bit, the host does not see that the device is connected. The device controller stays in the disconnected state until the software writes 1 to this bit.</li> <li>If the software attempts a connect after the soft disconnect or detects a disconnect event, it must set DCTL[8-5] to 5 before reasserting the RUN_STOP bit. The minimum duration of keeping this bit cleared is specified in <a href="#">Table 36-6</a>.</li> <li>When the USB or Link is in a lower power state and the Two Power Rails configuration is selected, software writes 0 to this bit to indicate that it is going to turn off the Core Power Rail. After the software turns on the Core Power Rail again and re-initializes the device controller, it must set this bit to start the device controller. For more details, see <a href="#">Low power operation</a></li> </ol>
30 CSFTRST	<p>Core Soft Reset</p> <p>Resets all clock domains as follows-</p> <ul style="list-style-type: none"> <li>Clears the interrupts and all the CSRs except the following registers: GCTL, GUCTL, GSTS, GUID, GUSB2PHYCFG, GUSB3PIPECTL, DCFG, DCTL, DEVTEN, DSTS</li> <li>All module state machines (except the SoC Bus Slave Unit) are reset to the IDLE state, and all the TxFIFOs and the Rx FIFO are flushed.</li> <li>Any transactions on the SoC bus Master are terminated as soon as possible, after gracefully completing the last data phase of a SoC bus transfer. Any transactions on the USB are terminated immediately.</li> </ul>

Table continues on the next page...

**USBx\_DCTL field descriptions (continued)**

Field	Description
	The application can write this bit at any time to reset the core. This is a self-clearing bit; the core clears this bit after all necessary logic is reset in the core, which may take several clocks depending on the core's current state. Once this bit is cleared, the software must wait at least 3 PHY clocks before accessing the PHY domain (synchronization delay). Typically, software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration registers listed above. When you change the PHY, the corresponding clock for the PHY is selected and used in the PHY domain. Once a new clock is selected, the PHY domain must be reset for proper operation.
29 —	This field is reserved.
28–24 HIRDTHRES	<p><b>HIRD Threshold</b></p> <p>The core asserts output signals utmi_l1_suspend_n and utmi_sleep_n on the basis of this signal.</p> <ul style="list-style-type: none"> <li>The core asserts utmi_l1_suspend_n to put the PHY into Deep Low-Power mode in L1 when both of the following are true:- HIRD value is greater than or equal to the value in DCTL[HIRD_Thres][3-0] <ul style="list-style-type: none"> <li>- HIRD_Thres[4] is set to 1'b1.</li> </ul> </li> <li>The core asserts utmi_sleep_n on L1 when one of the following is true:- If the HIRD value is less than HIRD_Thres[3-0] or <ul style="list-style-type: none"> <li>- HIRD_Thres[4] is set to 1'b0.</li> </ul> </li> </ul> <p><b>NOTE:</b> This field must be set to '0' during SuperSpeed mode of operation.</p>
23–20 LPM_NYET_thres	<p>When LPM Errata is enabled-</p> <p>Bits [23-20]- LPM NYET Response Threshold (LPM_NYET_thres) Handshake response to LPM token specified by device application. Response depends on DCFG[LPMCap].</p> <ul style="list-style-type: none"> <li>DCFG[LPMCap] is 1'b0 - The core always responds with Timeout (that is, no response).</li> <li>DCFG[LPMCap] is 1'b1 - The core responds with an ACK on successful LPM transaction, which requires that all of the following are satisfied:- There are no PID or CRC5 errors in both the EXT token and the LPM token (if not true, inactivity results in a timeout ERROR) <ul style="list-style-type: none"> <li>- A valid bLinkState = 0001B (L1) is received in the LPM transaction (else STALL) .</li> <li>- No data is pending in the Transmit FIFO and OUT endpoints not in flow controlled state (else NYET).</li> <li>- The BESL value in the LPM token is less than or equal to LPM_NYET_thres[3-0]</li> </ul> </li> </ul>
19–18 —	<p>This field is reserved.</p> <p><b>NOTE:</b> Software should program bit 19 to 0.</p>
17 CRS	<p><b>Controller Restore State</b></p> <p>This command is similar to the USBCMD[CRS] bit in host mode and initiates the restore process. When software sets this bit to '1', the controller immediately sets DSTS[RSS] to '1'. When the controller has finished the restore process, it sets DSTS[RSS] to '0'.</p> <p><b>NOTE:</b> When read, this field always returns '0'.</p>
16 CSS	<p><b>Controller Save State</b></p> <p>This command is similar to the USBCMD[CSS] bit in host mode and initiates the save process. When software sets this bit to '1', the controller immediately sets DSTS[SSS] to '1'. When the controller has finished the save process, it sets DSTS[SSS] to '0'.</p> <p><b>NOTE:</b> When read, this field always returns '0'.</p>
15–13 —	This field is reserved.

*Table continues on the next page...*

**USBx\_DCTL field descriptions (continued)**

Field	Description												
12 INITU2ENA	<p>Initiate U2 Enable</p> <p>0: May not initiate U2 (default)</p> <p>1: May initiate U2</p> <p>On USB reset, hardware clears this bit to "0". Software sets this bit after receiving SetFeature(U2_ENABLE), and clears this bit when ClearFeature(U2_ENABLE) is received.</p> <p>If DCTL[ACCEPTU2ENA] is 0, the link immediately exits U2 state.</p>												
11 ACCEPTU2ENA	<p>Accept U2 Enable</p> <p>0: Reject U2 except when Force_LinkPM_Accept bit is set (default)</p> <p>1: Core accepts transition to U2 state if nothing is pending on the application side.</p> <p>On USB reset, hardware clears this bit to "0". Software sets this bit after receiving a SetConfiguration command.</p>												
10 INITU1ENA	<p>Initiate U1 Enable</p> <p>0: May not initiate U1 (default)</p> <p>1: May initiate U1</p> <p>On USB reset, hardware clears this bit to "0". Software sets this bit after receiving SetFeature(U1_ENABLE), and clears this bit when ClearFeature(U1_ENABLE) is received.</p> <p>If DCTL[ACCEPTU1ENA] is 0, the link immediately exits U1 state.</p>												
9 ACCEPTU1ENA	<p>Accept U1 Enable</p> <p>0: Core rejects U1 except when Force_LinkPM_Accept bit is set (default)</p> <p>1: Core accepts transition to U1 state if nothing is pending on the application side.</p> <p>On USB reset, hardware clears this bit to "0". Software sets this bit after receiving a SetConfiguration command.</p>												
8–5 ULSTCHNGREQ	<p>USB/Link State Change Request</p> <p>Software writes this field to issue a USB/Link state change request. A change in this field indicates a new request to the core. If software wants to issue the same request back-to-back, it must write a 0 to this field between the two requests. The result of the state change request is reflected in the USB/Link State in DSTS. These bits are self-cleared on the MAC Layer exiting suspended state.</p> <p>If software is updating other fields of the DCTL register and not intending to force any link state change, then it must write a 0 to this field.</p> <p>SS Compliance mode is normally entered and controlled by the remote link partner. Refer to the USB3 specification. Alternatively, you can force the local link directly into Compliance mode, by resetting the SS link with the RUN_STOP bit set to zero. If you then write "10" to the USB/Link State Change field and "1" to RUN_STOP, the Link will go to Compliance. Once you are in Compliance, you may alternately write "zero" and "10" to this field to advance the compliance pattern.</p> <p>In SS mode-</p> <table border="1"> <thead> <tr> <th>Value</th><th>Requested Link State Transition/Action</th></tr> </thead> <tbody> <tr> <td>0</td><td>No Action</td></tr> <tr> <td>4</td><td>SS.Disabled</td></tr> <tr> <td>5</td><td>Rx.Detect</td></tr> <tr> <td>6</td><td>SS.Inactive</td></tr> <tr> <td>8</td><td>Recovery</td></tr> </tbody> </table>	Value	Requested Link State Transition/Action	0	No Action	4	SS.Disabled	5	Rx.Detect	6	SS.Inactive	8	Recovery
Value	Requested Link State Transition/Action												
0	No Action												
4	SS.Disabled												
5	Rx.Detect												
6	SS.Inactive												
8	Recovery												

*Table continues on the next page...*

**USBx\_DCTL field descriptions (continued)**

Field	Description	
	Value	Requested Link State Transition/Action
	Others	Reserved
In HS/FS/LS mode-		
	Value	Requested USB state transition
	8	Remote wakeup request
	Others	Reserved
The Remote wakeup request should be issued 2is after the device goes into suspend state (DSTS[21:18] is 3 - refer to <a href="#">Device Status Register (USB_DSTS)</a> ).		
4–1 TSTCTL	Test Control 0000: Test mode disabled 0001: Test_J mode 0010: Test_K mode 0011: Test_SE0_NAK mode 0100: Test_Packet mode 0101: Test_Force_Enable Others: Reserved	
0 —	This field is reserved.	

### 36.2.48 Device Event Enable Register (USBx\_DEVTEN)

This register controls the generation of Device-Specific events. If an enable bit is set to 0, the event will not be generated.

Address: Base address + C708h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### USBx\_DEVTEN field descriptions

Field	Description
31–14 —	This field is reserved.
13 —	This field is reserved.
12 VENDEVTSTRCDEN	Vendor Device Test LMP Received Event
11–10 —	This field is reserved.
9 ERRTICERREVTEN	Erratic Error Event Enable
8 —	This field is reserved.
7 SOFTEVTEN	Start of (micro)Frame Enable For debug purposes only; normally software must disable this event.

Table continues on the next page...

**USBx\_DEVTEN field descriptions (continued)**

Field	Description
6 U3L2L1SuspEn	U3/L2-L1 Suspend Event Enable
5 —	This field is reserved.
4 WKUPEVTEN	Resume/Remote Wakeup Detected Event Enable
3 ULSTCNGEN	USB/Link State Change Event Enable
2 CONNECTDNEEVVTEN	Connection Done Enable
1 USBRSTEVVTEN	USB Reset Enable
0 DISSCONNEVTEN	Disconnect Detected Event Enable

### 36.2.49 Device Status Register (USBx\_DSTS)

This register indicates the status of the device controller with respect to USB-related events.

Address: Base address + C70Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	0	0	RSS	SSS	COREIDLE	DEVCTRLHLT	USBLNKST	0	0	0	RXFIFOEMPTY	SOFF N
W	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

## USB Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R									SOFFN						CONNECTSPD		
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	1	0	0

### USBx\_DSTS field descriptions

Field	Description
31–30 —	This field is reserved.
29 —	This field is reserved.
28 —	This field is reserved.
27–26 —	This field is reserved.
25 RSS	Restore State Status This bit is similar to the USBSTS[RSS] in host mode. When the controller has finished the restore process, it will complete the command by setting DSTS[RSS] to '0'.
24 SSS	Save State Status This bit is similar to the USBSTS[SSS] in host mode. When the controller has finished the save process, it will complete the command by setting DSTS[SSS] to '0'.
23 COREIDLE	Core Idle The bit indicates that the core finished transferring all RxFIFO data to system memory, writing out all completed descriptors, and all Event Counts are zero. <b>NOTE:</b> While testing for Reset values, mask out the read value. This bit represents the changing state of the core and does not hold a static value.
22 DEVCTRLHLT	Device Controller Halted This bit is set to 0 when the Run/Stop bit in the DCTL register is set to 1.

Table continues on the next page...

**USBx\_DSTS field descriptions (continued)**

Field	Description
	The core sets this bit to 1 when, after software sets Run/Stop to '0', the core is idle and the lower layer finishes the disconnect process. When Halted =1, the core does not generate Device events.
21–18 USBLNKST	USB/Link State In SS mode: LTSSM State 4'h0 U0 4'h1 U1 4'h2 U2 4'h3 U3 4'h4 SS_DIS 4'h5 RX_DET 4'h6 SS_INACT 4'h7 POLL 4'h8 RECOV 4'h9 HRESET 4'ha CMPLY 4'hb LPBK 4'hf Resume/Reset In HS/FS/LS mode. 4'h0 On state 4'h2: Sleep (L1) state 4'h3 Suspend (L2) state 4'h4 Disconnected state (Default state) Software must write '8' (Recovery) to the DCTL[ULStChngReq] field to acknowledge the resume/reset request.
17 RXFIFOEMPTY	RxFIFO Empty
16–3 SOFFN	Frame/Microframe Number of the Received SOF When the core is operating at high-speed, [16:6] indicates the frame number [5:3] indicates the microframe number [16:14] is not used. Software can ignore these 3 bits [13:3] indicates the frame number
CONNECTSPD	Connected Speed Indicates the speed at which the USB 3.0 core has come up after speed detection through a chirp sequence. 100: SuperSpeed (PHY clock is running at 125 MHz or 250 MHz) 000 High-speed (PHY clock is running at 30 MHz or 60 MHz) 001 Full-speed (PHY clock is running at 30 MHz or 60 MHz) 010 Low-speed (PHY clock is running at 6 MHz)

*Table continues on the next page...*

**USBx\_DSTS field descriptions (continued)**

Field	Description
	011 Full-speed (PHY clock is running at 48 MHz) Low-speed is not supported for devices using a UTMI+ PHY.

### 36.2.50 Device Generic Command Parameter Register (USBx\_DGCMMDPAR)

This register indicates the device command parameter. This must be programmed before or along with the device command. The available device commands are listed in [Device Generic Command Register \(USB\\_DGCMD\)](#).

Address: Base address + C710h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R																																	
W																																	

Reset 0

**USBx\_DGCMMDPAR field descriptions**

Field	Description
PARAMETER	Parameter for Device Command

### 36.2.51 Device Generic Command Register (USBx\_DGCMD)

This register enables software to program the core using a single generic command interface to send link management packets and notifications. This register contains command, control, and status fields relevant to the current generic command, while the DGCMMDPAR register provides the command parameter.

Any commands that are not present in the following table are considered Reserved.

**Table 36-7. Device\_Generic\_Command\_Types**

Command	Description
02h	<p>Set Periodic Parameters</p> <p>Parameter[9:0] (SystemExitLatency): Software should set this to the same value programmed by the host through the Set SEL device request, in microseconds.</p> <p>The Set SEL control transfer has 6 bytes of data and contains 4 values.</p> <p>Offset Name Meaning</p> <p>0 U1SEL Time in <math>\mu</math>s for U1 System Exit Latency</p>

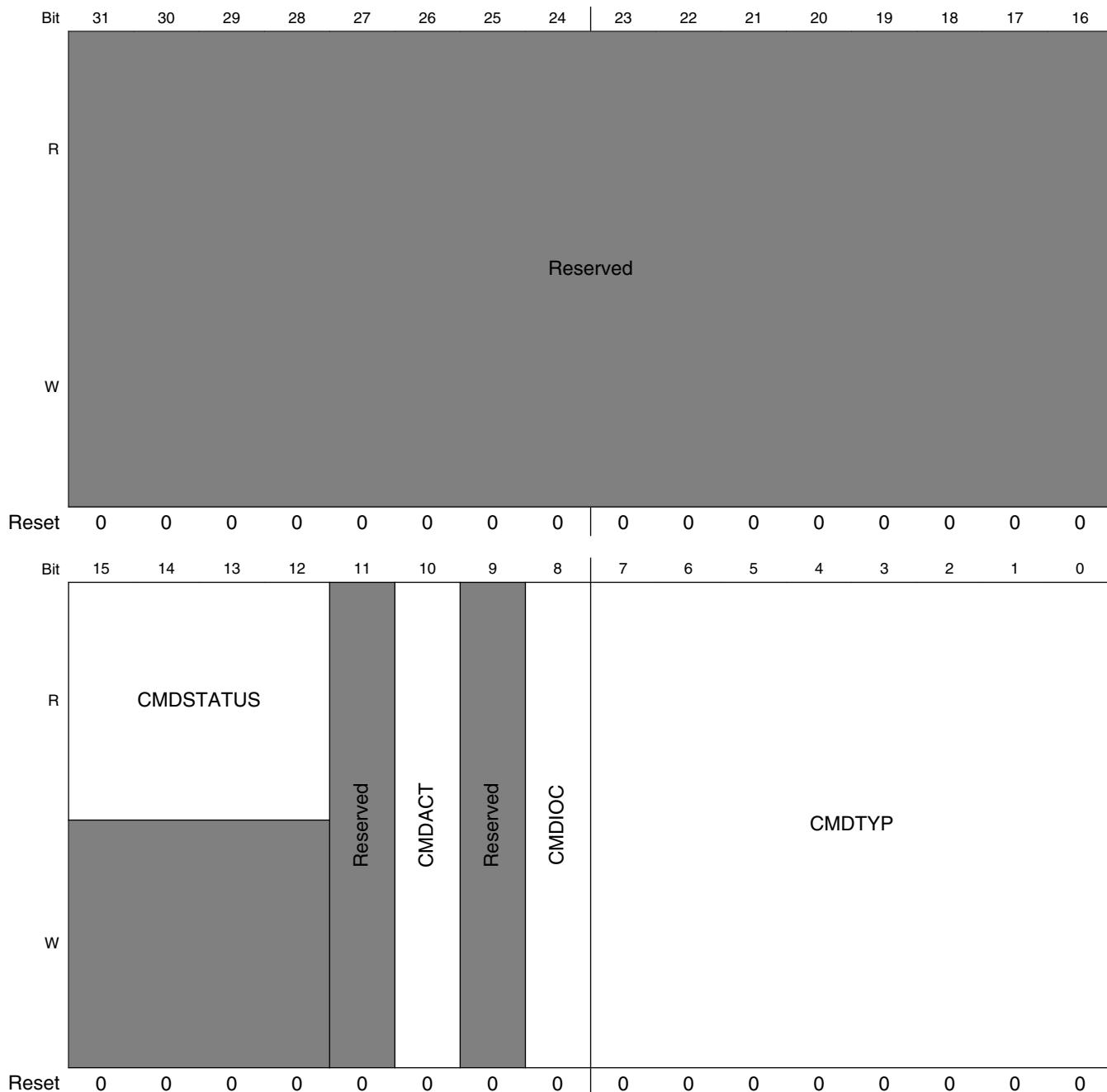
*Table continues on the next page...*

**Table 36-7. Device\_Generic\_Command\_Types (continued)**

Command	Description
	<p>1 U1PEL Time in <math>\mu</math>s for U1 Device to Host Exit Latency</p> <p>2 U2SEL Time in <math>\mu</math>s for U2 System Exit Latency</p> <p>4 U2PEL Time in <math>\mu</math>s for U2 Device to Host Exit Latency</p> <p>If the device is enabled for U1 and U2, then the U2PEL should be programmed. If the device is enabled only for U1, then U1PEL should be programmed into this parameter.</p> <p>If the value is greater than 125 <math>\mu</math>s, then the software must program a value of zero into this register.</p>
04h	<p>Set Scratchpad Buffer Array Address Lo</p> <p>This command sets bits [31:0] of the external address of the scratchpad buffer array used for save/restore.</p> <p>If either this command or command 05h is issued while the controller is stopped (RUN_STOP=0), the CmdIOC bit must be set to '0'.</p> <p>The device scratchpad buffer array has the same format as the xHCI scratchpad buffer array; it contains an array of 64-bit pointers to data buffers that will be used to save the controller's state.</p>
05h	<p>Set Scratchpad Buffer Array Address Hi</p> <p>This command sets bits [63:32] of the external address of the scratchpad array buffer used for save/restore.</p> <p>If either this command or command 04h is issued while the controller is stopped (RUN_STOP=0), the CmdIOC bit must be set to '0'.</p> <p>The device scratchpad buffer array has the same format as the xHCI scratchpad buffer array; it contains an array of 64-bit pointers to data buffers that will be used to save the controller's state.</p>
07h	<p>Transmit Device Notification</p> <p>This command allows any device notification to be transmitted, using the notification type and notification parameters specified in the DGCMMDPAR register.</p> <p>DGCMMDPAR[3:0] = Notification Type</p> <p>DGCMMDPAR[31:4] = Notification parameters, depends on the notification type</p> <p>For example, to transmit a Function Wake, software sets DGCMMDPAR[3:0] to 1, and DGCMMDPAR[10:4] to the Interface Number.</p> <p>This field relates to the "Notification Type Specific" field in a Device Notification Transaction Packet as described in Section 8.5.6 of the USB3 Specification. The following bits of the DGCMMDPAR register have been put into the corresponding DWORD described in Section 8.5.6 of the USB3 Specification:</p> <p>DGCMMDPAR[3:0] into DWORD 1[7:4] (Notification Type) DGCMMDPAR[27:4] into DWORD 1[31:8] (Notification Type Specific) DGCMMDPAR[31:28] into DWORD 2[3:0] (Notification Type Specific)</p> <p>There is one exception for the Bus Interval Adjustment Device Notification: DGCMMDPAR[19:4] represents the Bus Interval Adjustment field; however, in the USB3 specification, the bus interval adjustment field is actually at 31:16 of DWORD 1.</p>
09h	<p>Selected FIFO Flush</p> <p>Parameter[4:0] = FIFO Number</p> <p>Parameter[5] = '1' for TX FIFO or '0' for RX FIFO</p>
0Ah	<p>All FIFO Flush</p> <p>No Parameter</p>
0Ch	<p>Set Endpoint NRDY</p> <p>Issuing this command will make the core think that the given endpoint is in an NRDY state. If there are buffers available in that endpoint, the core will immediately transmit an ERDY.</p> <p>Parameter[4:0] = Physical Endpoint Number</p>

## USB Memory Map/Register Definition

Address: Base address + C714h offset



### USBx\_DGCMD field descriptions

Field	Description
31–16 —	This field is reserved.
15–12 CMDSTATUS	Command Status 1 CmdErr, indicates that the device controller encountered an error while processing the command. 0 Indicates command success

Table continues on the next page...

## USBx DGCMD field descriptions (continued)

Field	Description
11 —	This field is reserved.
10 CMDACT	<p>Command Active</p> <p>The software sets this bit to 1 to enable the device controller to execute the generic command.</p> <p>The device controller sets this bit to 0 after executing the command.</p>
9 —	This field is reserved.
8 CMDIOC	<p>Command Interrupt on Complete</p> <p>When this bit is set, the device controller issues a Generic Command Completion event after executing the command. Note that this interrupt is mapped to DCFG[IntrNum].</p> <p><b>NOTE:</b> This field must not set to '1' if the DCTL[RunStop] field is '0'.</p>
CMDTYP	<p>Command Type</p> <p>Specifies the type of command the software driver is requesting the core to perform.</p> <p>00h Reserved</p> <p>01h Set Endpoint Configuration - 64 or 96-bit Parameter</p> <p>02h Set Endpoint Transfer Resource Configuration - 32-bit Parameter</p> <p>03h Get Endpoint State - No Parameter Needed</p> <p>05h Clear Stall (see Set Stall) - No Parameter Needed</p> <p>06h Start Transfer - 64-bit Parameter</p> <p>07h Update Transfer - No Parameter Needed</p> <p>08h End Transfer - No Parameter Needed</p> <p>09h Start New Configuration - No Parameter Needed</p>

### **36.2.52 Device Active USB Endpoint Enable Register (USBx\_DALEPENA)**

This register indicates whether a USB endpoint is active in a given configuration or interface.

Address: Base address + C720h offset

**USBx\_DALEPENA field descriptions**

Field	Description
31–8 —	This field is reserved.
USBACTEP	<p>USB Active Endpoints</p> <p>This field indicates if a USB endpoint is active in the current configuration and interface.</p> <p>Bit[0] USB EP0-OUT</p> <p>Bit[1] USB EP0-IN</p> <p>Bit[2] USB EP1-OUT</p> <p>Bit[3] USB EP1-IN</p> <p>Bit[4] USB EP2-OUT</p> <p>Bit[5] USB EP2-IN</p> <p>Bit[6] USB EP3-OUT</p> <p>Bit[7] USB EP3-IN</p> <p>The entity programming this register must set bits 0 and 1 because they enable control endpoints that map to physical endpoints (resources) after USBReset.</p> <p>Hardware clears these bits for all endpoints (other than EP0-OUT and EP0-IN) after detecting a USB reset event. After receiving SetConfiguration and SetInterface requests, the application must program endpoint registers accordingly and set these bits.</p>

### 36.2.53 Device Physical Endpoint-n Command Parameter 2 Register (USBx\_DEPCMDPAR2n)

This register indicates the physical endpoint command Parameter 2. It must be programmed before issuing the command.

Address: Base address + C800h offset + (16d x i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

**USBx\_DEPCMDPAR2n field descriptions**

Field	Description
PARAMETER	Parameter 2

### 36.2.54 Device Physical Endpoint-n Command Parameter 1 Register (USBx\_DEPCMDPAR1n)

This register indicates the physical endpoint command Parameter 1. It must be programmed before issuing the command.

Address: Base address + C804h offset + (16d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_DEPCMDPAR1n field descriptions

Field	Description
PARAMETER	Parameter 1

### 36.2.55 Device Physical Endpoint-n Command Parameter 0 Register (USBx\_DEPCMDPAR0n)

This register indicates the physical endpoint command parameter 0. This must be programmed before or along with the command. For commands needing only one 32-bit parameter, this register must be programmed with the command register.

Address: Base address + C808h offset + (16d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																

Reset 0

#### USBx\_DEPCMDPAR0n field descriptions

Field	Description
PARAMETER	Parameter 0

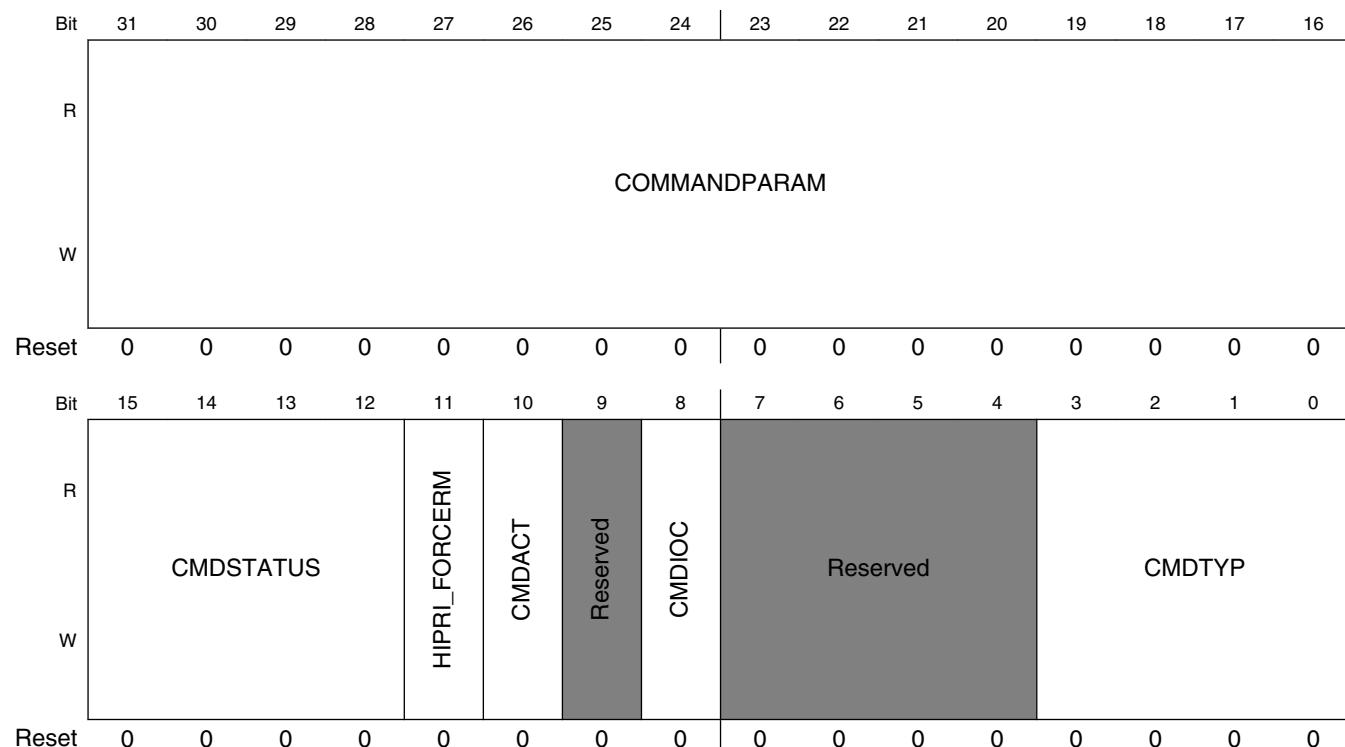
### 36.2.56 Device Physical Endpoint-n Command Register (USBx\_DEPCMDn)

This register enables software to issue physical endpoint-specific commands. This register contains command, control, and status fields relevant to the current generic command, while the DEPCMDPAR[2:0]n registers provide command parameters and return status information.

Several fields (including Command Type) are write-only, so their read values are undefined. After power-on, prior to issuing the first endpoint command, the read value of this register is undefined. In particular, the CmdAct bit may be set after power-on. In this case, it is safe to issue an endpoint command.

For more details about the commands, refer [Device physical endpoint-specific commands](#).

Address: Base address + C80Ch offset + (16d × i), where i=0d to 7d



**USBx\_DEPCMDn field descriptions**

Field	Description
31–16 COMMANDPARAM	Command Parameters. When this register is written for Start Transfer command: [31-16] StreamID. The USB StreamID assigned to this transfer Start Transfer command applied to an isochronous endpoint:

*Table continues on the next page...*

**USBx\_DEPCMDn field descriptions (continued)**

Field	Description
	<p>- [31-16] StartMicroFramNum Indicates the (micro) frame number to which the first TRB applies For Update Transfer, End Transfer, and Start New Configuration commands:</p> <p>[22-16] Transfer Resource Index (XferRscldx). The hardware-assigned transfer resource index for the transfer, which was returned in response to the Start Transfer command. The application software-assigned transfer resource index for a Start New Configuration command.</p> <p>Event Parameters (EventParam). When this register is read, refer to bits 31:16 in <a href="#">Table 36-16</a></p>
15-12 CMDSTATUS	<p>Command Completion Status</p> <p>Additional information about the completion of this command is available in this field. The information is in the same format as bits 15-12 of the Endpoint Command Complete event, refer <a href="#">Table 36-16</a></p>
11 HIPRI_FORCERM	<p>HighPriority/ForceRM</p> <p>HighPriority: Only valid for Start Transfer command</p> <p>ForceRM: Only valid for End Transfer command</p> <p>ClearPendIN: Only valid for Clear Stall command - Software sets this bit to clear any pending IN transaction (on that endpoint) stuck at the lower layers when a Clear Stall command is issued.</p>
10 CMDACT	<p>Command Active</p> <p>Software sets this bit to 1 to enable the device endpoint controller to execute the generic command. The device controller sets this bit to 0 when the CmdStatus field is valid and the endpoint is ready to accept another command. This does not imply that all the effects of the previously-issued command have taken place.</p>
9 —	This field is reserved.
8 CMDIOC	<p>Command Interrupt on Complete</p> <p>When this bit is set, the device controller issues a generic Endpoint Command Complete event after executing the command. Note that this interrupt is mapped to DEPCFG[IntrNum]. When the DEPCFG command is executed, the command interrupt on completion goes to the interrupt pointed by the DEPCFG[IntrNum] in the current command.</p> <p><b>NOTE:</b> This field must not set to '1' if the DCTL[RunStop] field is '0'.</p>
7-4 —	This field is reserved.
CMDTYP	<p>Command Type</p> <p>Specifies the type of command the software driver is requesting the core to perform.</p> <p>00h: Reserved</p> <p>01h: Set Endpoint Configuration</p> <ul style="list-style-type: none"> <li>- 64 or 96-bit Parameter</li> </ul> <p>02h: Set Endpoint Transfer Resource Configuration</p> <ul style="list-style-type: none"> <li>- 32-bit Parameter</li> </ul> <p>03h: Get Endpoint State</p> <ul style="list-style-type: none"> <li>- No Parameter Needed</li> </ul> <p>04h: Set Stall</p> <ul style="list-style-type: none"> <li>- No Parameter Needed</li> </ul> <p>05h: Clear Stall (see Set Stall)</p>

*Table continues on the next page...*

**USBx\_DEPCMDn field descriptions (continued)**

Field	Description
	<ul style="list-style-type: none"> <li>- No Parameter Needed</li> <li>06h: Start Transfer</li> <li>- 64-bit Parameter</li> <li>07h: Update Transfer</li> <li>- No Parameter Needed</li> <li>08h: End Transfer</li> <li>- No Parameter Needed</li> <li>09h: Start New Configuration</li> <li>- No Parameter Needed</li> </ul>

**36.2.57 OTG Configuration Register (USBx\_OCFG)**

This register specifies the HNP and SRP capability of the USB 3.0 core.

Address: Base address + CC00h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USBx\_OCFG field descriptions

Field	Description
31–6 —	This field is reserved.
5 DISPRTPWRCUTOFF	<p>OTG Disable Port Power Cut Off</p> <p>0- The core will automatically switch-off the VBUS by clearing the OCTL[PrtPwrCtl] after A_WAIT_BCON Timeout whenever the port is disconnected in disconnected state.</p> <p>1- The core will maintain VBUS ON even after A_WAIT_BCON Timeout when port is in disconnected state. The core will be in A_WAIT_BCON state continuously waiting for a Connect.</p>
4 —	This field is reserved.
3 OTGSFTRSTMSK	<p>OTG Soft Reset Mask</p> <p>This bit is used to mask specific soft resets from affecting the OTG functionality of the core. When set, the xHCI-based USBCMD[HCRST] in host mode and DCTL[CSFTRST] in device mode will be masked from affecting reset signal outputs sent to the PHY, the OTG FSM logic of the core and also the resets to the VBUS filters inside the core.</p> <p>0: The xHCI-based USBCMD[HCRST] and DCTL[CSFTRST] will reset the OTG logic of the core.</p> <p>1: The xHCI-based USBCMD[HCRST] and DCTL[CSFTRST] will be masked from the OTG logic of the core.</p> <p>This bit can be programmed to allow existing xHCI flows (with USBCMD[HCRST] programming) to function in OTG scenarios without any software changes.</p> <p>This bit should be programmed only when GCTL[PRTCAPDIR] = 2'b11. Otherwise it should be set at 1'b0.</p> <p><b>NOTE:</b> When using the core for OTG2 applications, it is not recommended to program USBCMD[HCRST] during role switch.</p>
2 —	This field is reserved.
1 HNPCAP	<p>HNP Capability</p> <p>The application uses this bit to control the USB 3.0 core's HNP capabilities.</p> <p>0: HNP capability is not enabled.</p> <p>1: HNP capability is enabled.</p>
0 SRPCAP	<p>SRP Capability</p> <p>The application uses this bit to control the USB 3.0 core's SRP capabilities.</p> <p>0: SRP capability is not enabled.</p> <p>1: SRP capability is enabled.</p> <p>If this bit is not set for B-device, it cannot request the connected A-device (host) to activate VBUS and start a session. If this bit is not set for A-device, it cannot detect the SRP from B-device (device) to activate VBUS and start a session.</p>

### 36.2.58 OTG Control Register (USBx\_OCTL)

The OTG Control register controls the behavior of the OTG function of the core.

Address: Base address + CC04h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved	PERIMODE	PRTPWRCTL	HNPREQ	SESREQ	TERMSEIDL PULSE	DEVSETHNPEN	HSTSETHNPEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBx\_OCTL field descriptions**

Field	Description
31–8 —	This field is reserved.
7 —	This field is reserved.
6 PERIMODE	Peripheral Mode Application uses this bit to program the core to work as a peripheral or as a host. 0: The OTG device acts as a host 1: The OTG device acts as a peripheral
5 PRTPWRCTL	Port Power Control Application sets this bit to initiate VBUS drive when it is an A- device. The application should clear this bit only if it wants to switch off the VBUS to B-device. The core clears this bit in the following conditions: Transition from any state to A-IDLE state defined in OTG 2.0 state machine. When AIDL_BDIS_TOUT occurs in A_SUSPEND When A_WAIT_BCON_TOUT occurs in A_WAIT_BCON Transition to any B- state defined in OTG2.0 state machine
4 HNPREQ	HNP Request 0: No HNP request 1: HNP request The application sets this bit to initiate a HNP request to the connected USB host. The application clears this bit by writing a 1'b0 when either of the following is detected- OEVT[OTGBDevBHostEndEvnt] OEVT[OTGBDevVBusChngEvnt]
3 SESREQ	Session Request 0: No session request 1: Session request The application sets this bit to initiate a session request on the USB. Writing 1'b1 to this field will trigger the core to send SRP (data line pulsing) on PHY interface. In the absence of OEVT[OTGBDevSessVldDetEvnt] after a session request, the application must wait for atleast TB_SRP_FAIL time (6 secs) before initiating another session request. This field returns 1'b0 when read.
2 TERMSELDLPULSE	TermSel DLine Pulsing Selection This bit selects utmi_termselect to drive data line pulse during SRP. 0: Data line pulsing using utmi_txvalid (default). 1: Data line pulsing using utmi_termsel.
1 DEVSETHNPEN	Device Set HNP Enable 0: HNP is not enabled in the application 1: HNP is enabled in the application The application sets this bit in HS/FS mode, when it successfully receives a SetFeature.SetHNPEnable command from the connected USB host.

*Table continues on the next page...*

**USBx\_OCTL field descriptions (continued)**

Field	Description
0 HSTSETHNPEN	<p>Host Set HNP Enable</p> <p>0: Host Set HNP is not enabled</p> <p>1: Host Set HNP is enabled</p> <p>The application sets this bit in the following scenario-</p> <p>In HS/FS mode, when it has successfully enabled HNP (using the SetFeature.SetHNPEnable command) from the connected device.</p>

### 36.2.59 OTG Events Register (USBx\_OEVT)

Any event set in this register will cause `otg_interrupt` signal to go high. Writing 1'b1 to the event information bit in this register clears the register bit and the associated interrupt. The `otg_interrupt` signal goes low when there are no more pending OTG events.

Address: Base address + CC08h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DeviceMode															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OTG events:

- OTGXhciRunStopSetEvt (bit 27)
- OTGDevRunStopSetEvt (bit 26)
- OTGConIDStsChngEvt (bit 25)
- OTGADevIdleEvt (bit 23)
- OTGADevBHostEndEvt (bit 20)
- OTGADevHostEvt (bit 19)
- OTGADevHNPChngEvt (bit 18)
- OTGADevSRPDetEvt (bit 17)
- OTGADevSessEndDetEvt (bit 16)

## USB Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
	Reserved				OTGBDevBHostEndEvt	OTGBDevHNPChngEvt	OTGBDevSessVldDteEvt	OTGBDevvBUSChngEvt					BSesVld	HstNegSs	SesReqSs	OEVTError
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBx\_OEVT field descriptions

Field	Description
31 DeviceMode	Device Mode Indicates whether the device is in A-device or B-device mode based on utmiotg_iddig 0: A-Device mode 1: B-Device mode The rest of the OTG Event Information bits (OTGxxxxEvtInfo) in OEVT register will be based on the contents of this field.
30–28 —	This field is reserved.
27 OTGXhciRunStpSetEvt	OTG Host Run Stop Set Event This event is set when the Host Driver programs the USBCMD[Run/Stop] to 1'b1.
26 OTGDevRunStpSetEvt	OTG Device Run Stop Set Event This event is set when the Device Driver programs the DCTL[Run/Stop] to 1'b1.
25 —	This field is reserved.
24 OTGConIDstsChngEvt	Connector ID Status Change Event

Table continues on the next page...

**USBx\_OEVT field descriptions (continued)**

Field	Description
	Set in both A-Dev/B-Dev Mode. This event is generated when there is a change in connector ID status.
23–22 —	This field is reserved.
21 OTGADevIdleEvt	A-device A-IDLE Event Set in A-device Mode Only. The event is generated when A- device enters A-IDLE state. This event is set when the OTG 2.0 FSM of the core enters A-IDLE state from any other OTG state.
20 OTGADevBHostEndEvt	A-device B-Host End Event Set in A-device Mode Only. The event is generated when B- device has completed its host role. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
19 OTGADevHostEvt	A- device host event Set in A-device Mode Only. This event is generated when A- device enters host role. In HS/FS mode, it occurs after the initial connect to a B- device as A-host as well as when there is a role change from A-peripheral to A-host. <b>NOTE:</b> This bit is applicable only for OTG 2.0 mode of operation.
18 OTGADevHNPChngEvt	A-Dev HNP Change Event Set in A-device Mode Only. The event is generated when there is an HNP attempt. <b>NOTE:</b> This bit is applicable only for OTG 2.0 mode of operation.
17 OTGADevSRPDetEvt	SRP Detect Event Set in A-device Mode Only. This event is asserted when a session request from the B-device is detected via SRP. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
16 OTGADevSessEndDetEvt	Session End Detected Event Set in A-device Mode Only. This event is asserted when the utmiotg_vbusvalid signal goes low indicating the end of a session. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
15–12 —	This field is reserved.
11 OTGBDevBHostEndEvt	B-Device B-Host End Event Set in B-device Mode Only. This event is generated when B- device has completed its host role. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
10 OTGBDevHNPChngEvt	B-Dev HNP Change Event Set in B-Device Mode only. This event is generated when there is a Success or Failure of an HNP attempt. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
9 OTGBDevSessVldDetEvt	Session Valid Detected Event Set in B-device Mode Only. This event is asserted when there is a valid VBUS from A-device and B-device succeeds in starting a session.

*Table continues on the next page...*

**USBx\_OEVT field descriptions (continued)**

Field	Description
	<b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
8 OTGBDevVBUSChngEvnt	VBUS Change Event Set in B-device Mode Only. This event is asserted when the utmisrp_bvalid signal goes low (indicating the end of a session), or goes high. <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
7–4 —	This field is reserved.
3 BSesVld	B-Session Valid Indicates the Device mode transceiver status. The core updates this bit when OEVTEN[OTGBDevVBUSChngEvnt] is set. 0: B-session is not valid 1: B-session is valid <b>NOTE:</b> This bit is applicable for OTG 2.0 mode of operation.
2 HstNegSts	Host Negotiation Status The core updates this bit when any of the following bits is set- OEVTEN[OTGADevHNPChngEvnt] OEVTEN[OTGBDevHNPChngEvnt] This bit indicates Host Negotiation Success or Failure. 0: Host negotiation failure. In A-device, for HS/FS, this indicates an imminent end of session indication from the core. In B-device, for HS/LS, it indicates that the timer used to wait for an A-device to signal a connection (b_ase0_burst_tmout in OTG 2.0) timed out resulting in B-device staying as B-peripheral. 1: Host negotiation success. This indicates that the host negotiation was successful. <b>NOTE:</b> This bit is applicable only for OTG 2.0 mode of operation.
1 SesReqSts	Session Request Status Ignore this field.
0 OEVTError	OTG Event Error There are no errors currently defined.

### 36.2.60 OTG Events Enable Register (USBx\_OEVTEM)

Setting a bit in this register enables the generation of corresponding events in OEVT and assertion of otg\_interrupt due to this event. When the enable bit is 1'b0, the event will not be set in OEVT and otg\_interrupt will not be asserted due to this event.

Address: Base address + CC0Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved				OTGXhciRunStpSetEvntEn	n	OTGDevRunStpSetEvntEn	n	Reserved	OTGConIDStsChngEvntEn	n	HRRConfNotifEvntEn	HRRIinitNotifEvntEn	OTGADevIdleEvntEn	OTGADevBHostEndEvntEn	n	OTGADevHostEvntEn
W					0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	Reserved				OTGBDevBHostEndEvntEn	n	OTGBDevHNPChngEvntEn	n	OTGBDevSessVidDetEvntEn	OTGBDevVBusChngEvntEn	n	Reserved					
W					0	0	0	0	0	0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### USBx\_OEVTEM field descriptions

Field	Description
31–28 —	This field is reserved.
27 OTGXhciRunStpSetEvntEn	OTG Host Run Stop Set Event Enable When this bit is set, OEVT[XhciRunStpSet] event is enabled. If not, that event is disabled.
26 OTGDevRunStpSetEvntEn	OTG Device Run Stop Set Event Enable When this bit is set, OEVT[DevRunStpSet] event is enabled. If not, that event is disabled.
25 —	This field is reserved.
24 OTGConIDStsChngEvntEn	OTGCommonEvtInfoEn[0] Connector ID Status Change Event Enable (OTGConIDStsChngEvntEn).

Table continues on the next page...

**USBx\_OEVTEM field descriptions (continued)**

Field	Description
	When this bit is set, OEVT[OTGConIDSChngEvnt] is enabled. If not, the event is disabled.
23 HRRConfNotifEvntEn	OTGCommonEvtInfoEn[2] HRRConfNotif Event Enable (HRRConfNotifEvntEn). When this bit is set, OEVT[HRRConfNotifEvnt] is enabled. If not, the event is disabled.
22 HRRInitNotifEvntEn	OTGCommonEvtInfoEn[1] HRRInitNotif Event Enable (HRRInitNotifEvntEn). When this bit is set, OEVT[HRRInitNotifEvnt] is enabled. If not, the event is disabled.
21 OTGADevIdleEvntEn	OTGADevEvtInfoEn[5] A-device A-IDLE Event (OTGADevIdleEvntEn) When this bit is set, OEVT[OTGADevIdleEvnt] is enabled. If not, the event is disabled.
20 OTGADevBHostEndEvntEn	OTGADevEvtInfoEn[4] A-device B-Host End Event Enable (OTGADevBHostEndEvntEn) When this bit is set, OEVT[OTGADevBHostEndEvnt] is enabled. If not, the event is disabled.
19 OTGADevHostEvntEn	OTGADevEvtInfoEn[3] A-device host event (OTGADevHostEvntEn) When this bit is set, OEVT[OTGADevHostEvnt] is enabled. If not, the event is disabled
18 OTGADevHNPChngEvntEn	OTGADevEvtInfoEn[2] A-Dev HNP Change EventEn (OTGADevHNPChngEvntEn) When this bit is set, OEVT[OTGADevHNPChngEvnt] is enabled. If not, the event is disabled
17 OTGADevSRPDetEvntEn	OTGADevEvtInfoEn[1] SRP Detect Event Enable (OTGADevSRPDetEvntEn) When this bit is set, OEVT[OTGADevSRPDetEvnt] is enabled. If not, the event is disabled
16 OTGADevSessEndDetEvntEn	OTGADevEvtInfoEn[0] Session End Detected Event Enable (OTGADevSessEndDetEvntEn) When this bit is set, OEVT[OTGADevSessEndEvnt] is enabled. If not, the event is disabled
15–12 —	This field is reserved.
11 OTGBDevBHostEndEvntEn	OTGBDevEvtInfoEn[3] B-device B-Host End Event Enable (OTGBDevBHostEndEvntEn) When this bit is set, OEVT[OTGBDevHostEndEvnt] is enabled. If not, the event is disabled
10 OTGBDevHNPChngEvntEn	OTGBDevEvtInfoEn[2] B-Dev HNP Change Event Enable (OTGBDevHNPChngEvntEn) When this bit is set, OEVT[OTGBDevHNPChngEvnt] is enabled. If not, the event is disabled
9 OTGBDevSessVldDetEvntEn	OTGBDevEvtInfoEn[1] Session Valid Detected Event Enable (OTGBDevSessVldDetEvntEn)

*Table continues on the next page...*

**USBx\_OEVTE field descriptions (continued)**

Field	Description
	Set in B-device Mode Only- This event is asserted when there is a valid VBUS from A- device and B-device succeeds in starting a session.
8 OTGBDevVBUSChngEvntEn	OTGBDevEvtInfoEn[0] VBUS Change Event Enable (OTGBDevVBUSChngEvntEn) When this bit is set, OEVT[OTGBDevVBUSChngEvnt] is enabled. If not, the event is disabled
—	This field is reserved.

### 36.2.61 OTG Status Register (USBx\_OSTS)

The OTG Status Register reflects the status of the OTG function of the core.

Address: Base address + CC10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

**USBx\_OTS field descriptions**

Field	Description
31–14 —	This field is reserved.
13 DevRunStp	This bit reflects the status of the device Run/Stop bit in the DCTL Device register. 0 Device Run/Stop is set to 0. 1 Device Run/Stop is set to 1.
12 xHciRunStp	OTG Host Run Stop Set Event This event is set when the Host Driver programs the [USBCMD[Run/Stop]] to 1'b1.
11–8 —	This field is reserved.
7–5 —	This field is reserved.
4 PeripheralState	Indicates whether the core is acting as a peripheral or host. 0: Host 1: Peripheral
3 xHCIPrtPower	This bit reflects the PORTSC[PP] bit in the xHCI register.
2 BSesVld	B-Session Valid Indicates the Device mode transceiver status. In OTG mode, applications use this bit to determine if the device is connected. 0: B-session is not valid. 1: B-session is valid.
1 ASesVld	VBUS Valid Indicates the Host mode transceiver status. 0: A-session is not valid 1: A-session is valid
0 ConIDsts	Connector ID Status It indicates the connector ID status. 0: The USB 3.0 core is in A-device mode 1: The USB 3.0 core is in B-device mode <b>NOTE:</b> The reset value of this field depends on the power-on value of the IDDIG signal from the PHY.

### 36.2.62 ADP Configuration Register (USBx\_ADPCFG)

Address: Base address + CC20h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PrbPer		PrbDelta		PrbDschg											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

#### USBx\_ADPCFG field descriptions

Field	Description
31–30 PrbPer	<p>Probe Period</p> <p>These bits set the T_ADP_PRB as follows-</p> <ul style="list-style-type: none"> <li>00: 0.775 sec</li> <li>01: 1.55 sec</li> <li>10: 2.275 sec</li> <li>11: Reserved</li> </ul> <p>The scaledown values for PrbPer are-</p> <ul style="list-style-type: none"> <li>00: 12.5ms</li> <li>01: 18.75ms</li> <li>10: 25 ms</li> <li>11: 31.25 ms</li> </ul>
29–28 PrbDelta	<p>Probe Delta</p> <p>These bits set the resolution for RTIM value. They are defined in units of 32 kHz clock cycles as follows-</p> <ul style="list-style-type: none"> <li>00: 1 cycles</li> <li>01: 2 cycles</li> <li>10: 3 cycles</li> <li>11: 4 cycles</li> </ul> <p>For example, if this value is chosen to be 2'b01, it means that RTIM increments for every two 32 kHz clock cycles.</p>
27–26 PrbDschg	<p>Probe Discharge</p> <p>These bits set the time for TADP_DSCHG. They are defined as follows-</p> <ul style="list-style-type: none"> <li>00: 4 msec</li> <li>01: 8 msec</li> <li>10: 16 msec</li> <li>11: 32 msec</li> </ul> <p>The scaledown values for the PrbDschg are as follows-</p> <ul style="list-style-type: none"> <li>00: 62.5 us</li> </ul>

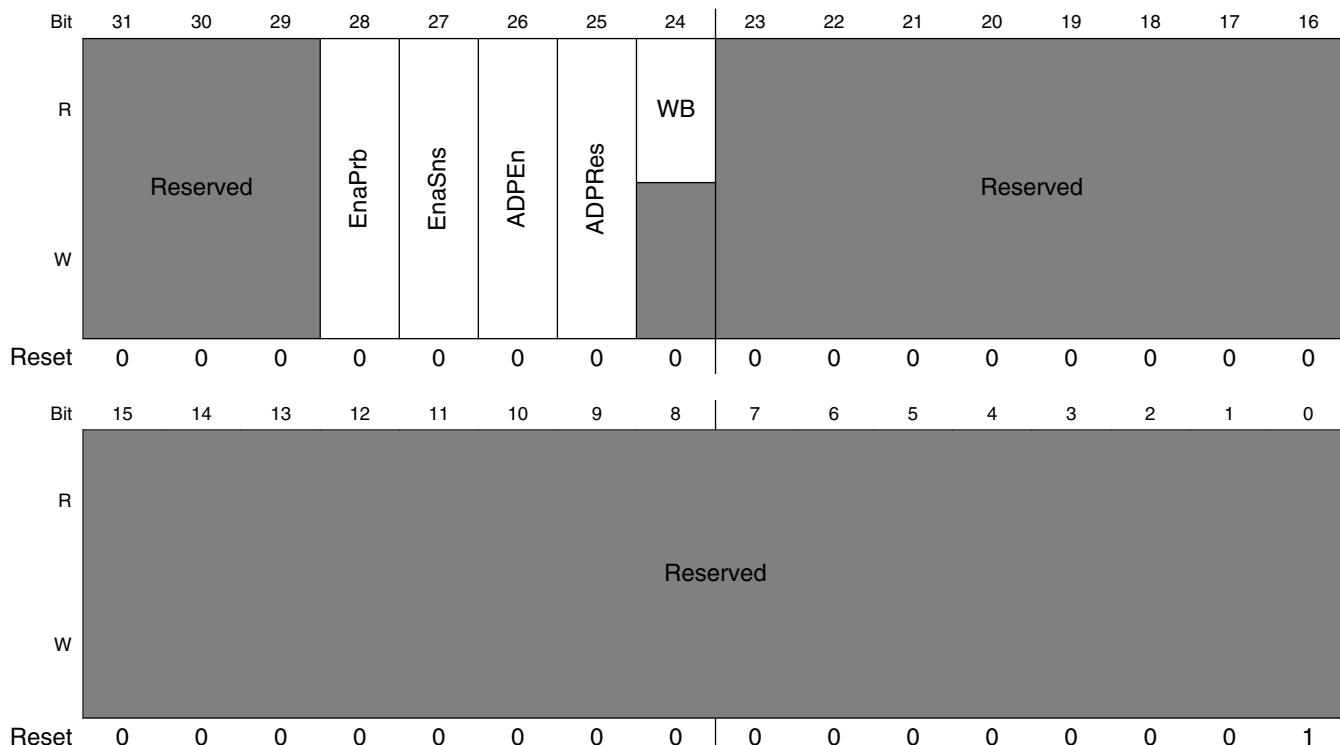
*Table continues on the next page...*

**USBx\_ADPCFG field descriptions (continued)**

Field	Description
	01: 125 us 10: 250 us 11: 500 us
—	This field is reserved.

**36.2.63 ADP Control Register (USBx\_ADPCTL)**

Address: Base address + CC24h offset

**USBx\_ADPCTL field descriptions**

Field	Description
31–29	This field is reserved.
—	
28 EnaPrb	Enable Probe When set to 1'b1 along with ADPEn, the core performs a probe operation.
27 EnaSns	Enable Sense When set to 1'b1 along with ADPEn, the core performs a sense operation.
26 ADPEn	ADP Enable When set to 1'b1, the core performs either ADP probing or sensing based on EnaPrb and EnaSns. ADPEn = 1'b0 gates the suspend clock for major portion of ADP related logic.

*Table continues on the next page...*

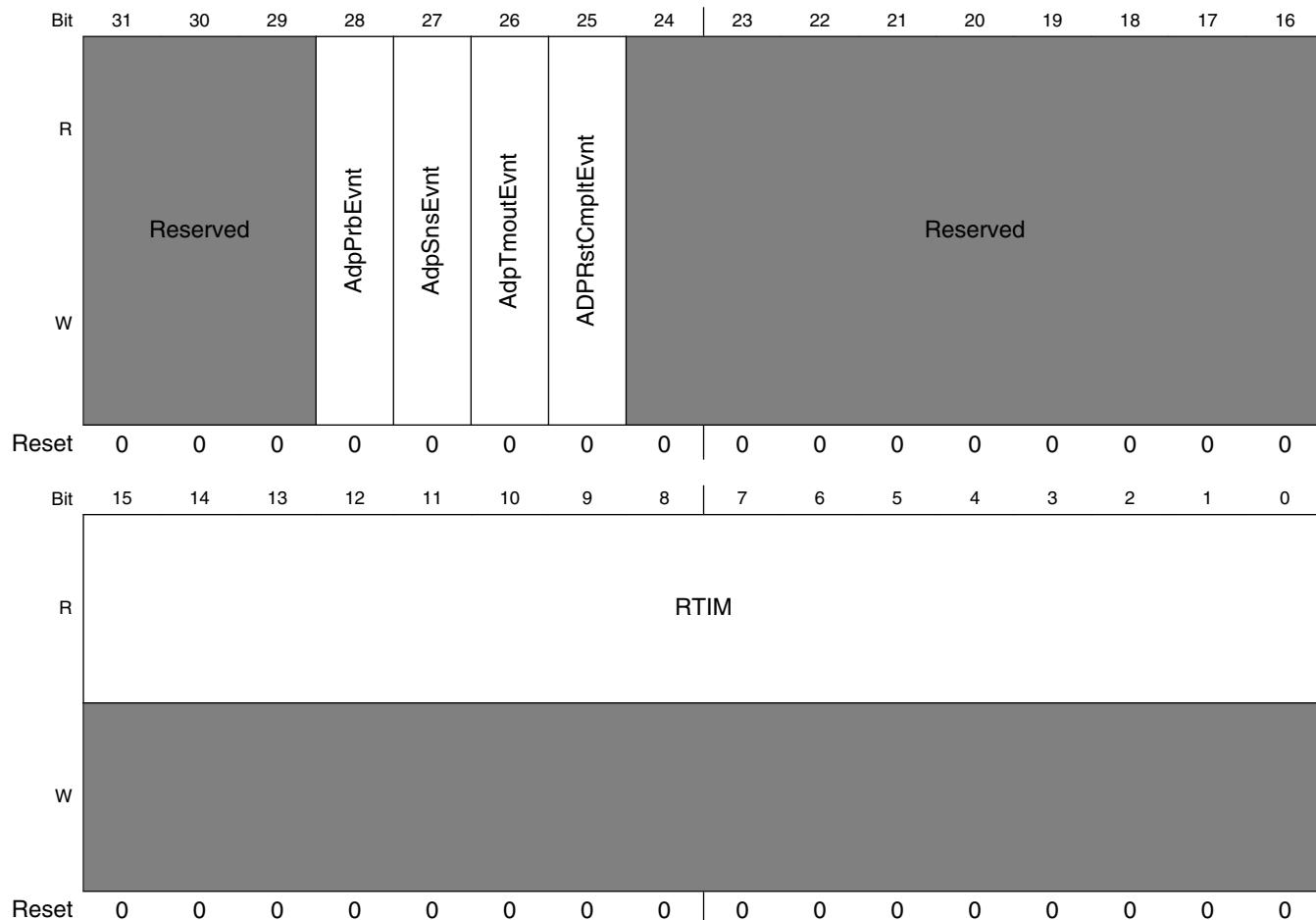
**USBx\_ADPCTL field descriptions (continued)**

Field	Description
25 ADPRes	ADP Reset When set to 1'b1, the ADP controller is reset. This bit is auto-cleared after the reset procedure is complete in the ADP controller.
24 WB	Write Busy 0: Write Completed 1: Write in Progress The application can read or write ADPCFG and ADPCTL registers only if this field is cleared. Hardware sets this bit when the write is in progress in the Suspend clock domain.
—	This field is reserved.

### 36.2.64 ADP Event Register (USBx\_ADPEVT)

Writing 1 to the information bit in this register clears the register bit and associated interrupt.

Address: Base address + CC28h offset



**USBx\_ADPEVT field descriptions**

Field	Description
31–29 —	This field is reserved.
28 AdpPrbEvt	ADPEVTInfo[4] ADP Probe Event (AdpPrbEvt) When this event is set, it means that the VBUS voltage is greater than VadpPrb or VadpPrb is reached.
27 AdpSnsEvt	ADPEVTInfo[3] ADP Sense Event (AdpSnsEvt)

*Table continues on the next page...*

**USBx\_ADPEVT field descriptions (continued)**

Field	Description
	When this event is set, it means that the VBUS voltage is greater than VadpSns or VadpSns is reached.
26 AdpTmoutEvt	ADP Timeout Event This event is relevant when ADP probe command is executed. When this event is set, it means that the ramp time is completed (GADPCTL.RTIM has reached its terminal value of 0x7FF). This is a debug feature that allows software to read the ramp time after each cycle.
25 ADPRstCmpltEvt	ADP Reset complete Event This event when set, indicates that the ADP Reset command is successful.
24–16 —	This field is reserved.
RTIM	RAMP TIME These bits capture the latest time it took for VBUS to ramp from VADP_SINK to VADP_PRB. The bits are defined in units of 32 kHz clock cycles as follows:  0x000: 1 cycles 0x001: 2 cycles 0x002: 3 cycles and so on till, 0xFFFF: 65536 cycles  The maximum time of 65536 cycles corresponds to a time of 2.04 seconds.  Note for scaledown ramp_timeout PrbDelta = 2'b00 => 6250 us PrbDelta = 2'b01 => 3125 us PrbDelta = 2'b10 => 1562.5 us PrbDelta = 2'b11 => 781.25 us

### 36.2.65 ADP Event Enable Register (USBx\_ADPEVTEN)

Address: Base address + CC2Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					AdpPrbEvntEn	AdpSnsEvntEn	AdpTmoutEvntEn	ADPRstCmpltEvntEn								
W					Reserved											
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

#### USBx\_ADPEVTEN field descriptions

Field	Description
31–29 —	This field is reserved.
28 A dpPrbEvntEn	ADP Probe Event Enable When this bit is set, ADPPrbEvnt in ADPEVT register is enabled.
27 A dpSnsEvntEn	ADP Sense Event Enable When this bit is set, A dpSnsEvntEn in ADPEVT register is enabled.
26 A dpTmoutEvntEn	ADP Timeout Event Enable When this bit is set, A dpTmoutEvntEn in ADPEVT register is enabled.
25 ADPRstCmpltEvntEn	ADP Reset complete Event Enable When this bit is set, ADPRstCmpltEvnt in ADPEVT register is enabled.
—	This field is reserved.

## 36.3 USB PHY Memory Map/Register Definition

### USB\_PHY\_SS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
84F_0000	SUP_IDCODE_LO (USB_PHY_SS1_IP_IDCODE_LO)	16	R	64CDh	<a href="#">36.3.1/2372</a>

Table continues on the next page...

**USB\_PHY\_SS memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
84F_0002	SUP_IDCODE_HI (USB_PHY_SS1_SUP_IDCODE_HI)	16	R	A21Ch	<a href="#">36.3.2/2372</a>
84F_0060	MPLL_LOOP_CTL (USB_PHY_SS1_MPLL_LOOP_CTL)	16	R/W	00C0h	<a href="#">36.3.3/2373</a>
84F_200C	LANE0_RX_OVRD_IN_HI (USB_PHY_SS1_LANE0_RX_OVRD_IN_HI)	16	R/W	0000h	<a href="#">36.3.4/2373</a>
850_0000	SUP_IDCODE_LO (USB_PHY_SS2_IP_IDCODE_LO)	16	R	64CDh	<a href="#">36.3.1/2372</a>
850_0002	SUP_IDCODE_HI (USB_PHY_SS2_SUP_IDCODE_HI)	16	R	A21Ch	<a href="#">36.3.2/2372</a>
850_0060	MPLL_LOOP_CTL (USB_PHY_SS2_MPLL_LOOP_CTL)	16	R/W	00C0h	<a href="#">36.3.3/2373</a>
850_200C	LANE0_RX_OVRD_IN_HI (USB_PHY_SS2_LANE0_RX_OVRD_IN_HI)	16	R/W	0000h	<a href="#">36.3.4/2373</a>
851_0000	SUP_IDCODE_LO (USB_PHY_SS3_IP_IDCODE_LO)	16	R	64CDh	<a href="#">36.3.1/2372</a>
851_0002	SUP_IDCODE_HI (USB_PHY_SS3_SUP_IDCODE_HI)	16	R	A21Ch	<a href="#">36.3.2/2372</a>
851_0060	MPLL_LOOP_CTL (USB_PHY_SS3_MPLL_LOOP_CTL)	16	R/W	00C0h	<a href="#">36.3.3/2373</a>
851_200C	LANE0_RX_OVRD_IN_HI (USB_PHY_SS3_LANE0_RX_OVRD_IN_HI)	16	R/W	0000h	<a href="#">36.3.4/2373</a>

**36.3.1 SUP\_IDCODE\_LO (USB\_PHY\_SSx\_IP\_IDCODE\_LO)**

Address: Base address + 0h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read	DATA															
Write																
Reset	0	1	1	0	0	1	0	0	1	1	0	0	1	1	0	1

**USB\_PHY\_SSx\_IP\_IDCODE\_LO field descriptions**

Field	Description
0–15 DATA	These bits indicate the IP version.

**36.3.2 SUP\_IDCODE\_HI (USB\_PHY\_SSx\_SUP\_IDCODE\_HI)**

Address: Base address + 2h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read	DATA															
Write																
Reset	1	0	1	0	0	0	1	0	0	0	0	1	1	1	0	0

**USB\_PHY\_SSx\_SUP\_IDCODE\_HI field descriptions**

Field	Description
0–15 DATA	These bits indicate the IP version.

**36.3.3 MPLL\_LOOP\_CTL (USB\_PHY\_SSx\_MPLL\_LOOP\_CTL)**

Address: Base address + 60h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read																
Write																

Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
-------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

**USB\_PHY\_SSx\_MPLL\_LOOP\_CTL field descriptions**

Field	Description
0–7 -	This field is reserved.
8–11 PROP_CNTRL	Charge pump proportional current setting
12–15 -	This field is reserved.

**36.3.4 LANE0\_RX\_OVRD\_IN\_HI  
(USB\_PHY\_SSx\_LANE0\_RX\_OVRD\_IN\_HI)**For initialization details, refer [Initialization/application information](#).

Address: Base address + 200Ch offset

Bit	0	1	2	3	4	5	6	7
Read								
Write								
					RX_EQ_OVRD			
Reset	0	0	0	0	0	0	0	0
Bit	8	9	10	11	12	13	14	15
Read	RX_EQ_EN_OVRD	RX_EQ_EN	RX_RATE_OVRD					
Write								
Reset	0	0	0	0	0	0	0	0

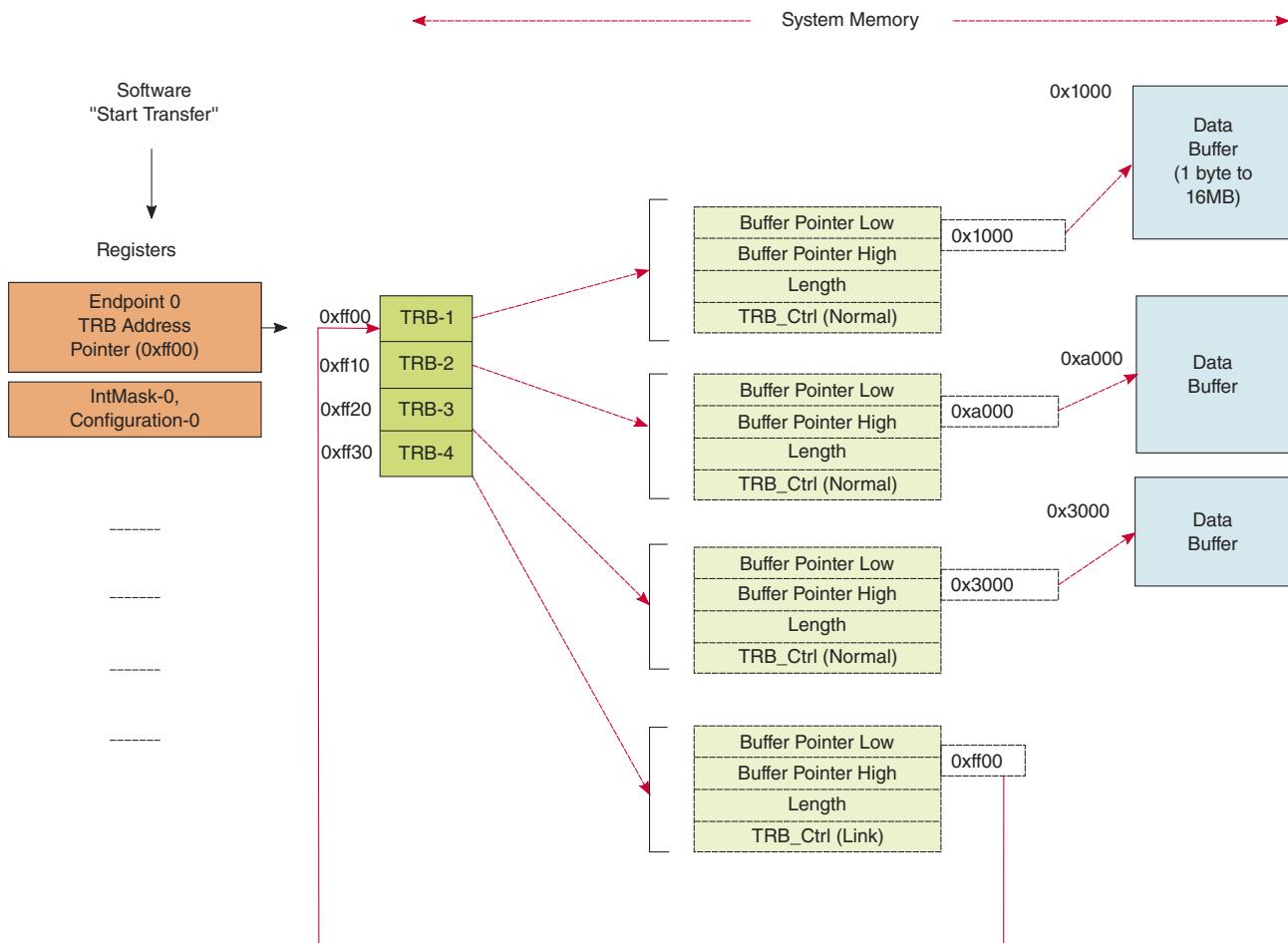
**USB\_PHY\_SSx\_LANE0\_RX\_OVRD\_IN\_HI field descriptions**

Field	Description
0–3 -	This field is reserved.
4 RX_EQ_OVRD	Override value for rx_eq
5–7 RX_EQ	Override value for rx_eq
8 RX_EQ_EN_ OVRD	Override enable for rx_eq_en
9 RX_EQ_EN	Override value for rx_eq_en
13 RX_RATE_ OVRD	Override enable for rx_rate
10–15 -	This field is reserved.

## 36.4 Functional Description

### 36.4.1 System memory descriptor and data buffers

The software creates transfer request blocks (TRBs), and four DWords each, that point to the data buffers. Normally, the TRBs are allocated consecutively in system memory; only the data buffers can be scattered. In the case of a circular buffer, the link-TRB points to the next TRB. Once the TRBs and data buffers are set up in the system memory, the software driver issues a start transfer command that points to the location of the first TRB in the system memory to start the DMA operation. TRBs, though small (only four DWords), provide a rich set of features for the software to schedule transfers, isochronous, control, interrupt moderation, and so on.



**Figure 36-4. System Memory Descriptor and Data Buffers**

### 36.4.2 Device descriptor structures

Device mode transfer request blocks (TRBs) are small 4 DWORDs and at the same time provide a rich set of features so that the software can efficiently manage the USB core, memory buffers, and MIPS requirements.

The following is a list of device mode TRB characteristics:

- TRBs provide support for scattered data structures. The scattered data buffers can be zero bytes to 16 MB in length.
- TRBs must be placed in system memory aligned to a 16-byte boundary.
- TRBs are kept in linear memory to enhance descriptor caching performance. If the data buffers are small (as in an Ethernet application) the core can efficiently collect the scattered data to build USB packets without wasting bus efficiency that collecting scattered descriptors requires.

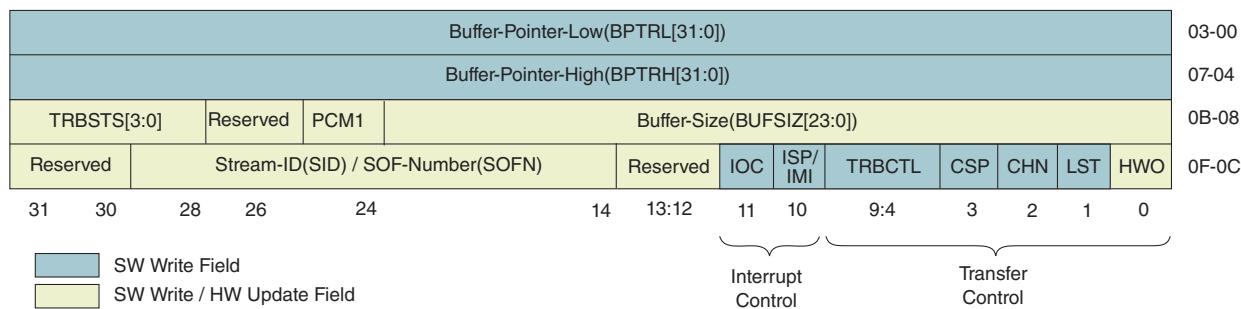
- Supports TRB linking. This allows software to set up a circular ring of TRBs with the last TRB linking to the first one.
- Supports system memory interface with 40-bit addressing capability
- Supports byte-aligned buffers for each TRB of a transfer. This feature prevents the need for buffer copying in cases where the USB device driver receives unaligned data buffers from other applications (for example, Ethernet). Whenever the application controls buffer allocation, it must allocate SoC bus width and burst-aligned buffers to facilitate efficient bus and memory utilization. For transmitting, the core supports byte-aligned buffers on all TRBs.

For example: If you plan to use 16 bursts in a 64-bit system, you must try to allocate buffers that are  $16 * 8 = 128$  bytes aligned. This is the normal buffer structure because Linux-like OSs allocate 4 KB buffers. SDR/DDR memory controllers also provide better performance when requests are burst aligned.

- Supports software queueing of multiple USB transfers (LST bit for IN/OUT transfers and CSP bit for OUT transfers control this function).
- Provides interrupt moderation capability, allowing software to selectively enable events on TRB completion, as controlled by the IOC (Interrupt on completion) bit. The IOC event is also used by software to reallocate the released buffers, allowing software to re-use just a few buffers in a circular fashion. This helps when the memory is limited but have enough MIPS to process interrupts. Larger buffers can be allocated to reduce the number of interrupts.
  - In USB 3.0 larger transfers are recommended because the raw transfer rate is almost 10 times faster, unlike USB 2.0, where drivers set up only 64 KB or 128 KB transfers. For example, a 64 KB transfer that takes 1.3 ms in USB 2.0 requires only 164 s in USB 3.0. If you set up buffers of 64 KB and enable the transfer completion event, then you receive an interrupt every 164 s.
- Supports streaming (Stream ID field used for this purpose).

### **36.4.2.1 Structures**

This figure shows the control and status field of a transfer request block.

**Figure 36-5. TRB Control and Status Fields****Table 36-8. Device Descriptor Structure Field Definitions**

Field	Description	Hardware Access
<b>DW 03-00</b>		
31:0	Buffer Pointer Low (BPTRL) Data buffer pointer to low 32 bit address (BPTR[31:0]). Hardware may also update this field (implementation specific).	R/W
<b>DW 07-04</b>		
31:0	Buffer Pointer High (BPTRH) Data buffer pointer to high 32-bit address (BPTR[63:32]).	R/W
<b>DW 0B-08</b>		
31:28	TRB Status (TRBSTS) Hardware updates this field with transfer status information before releasing the TRB. 4'h0: OK 4'h1: MissedIsoc: Isochronous interval missed or incomplete 4'h2: SetupPending - During the current control transfer data/status phase, another SETUP was received. 4'h4: TransferInProgress - During the current transfer, an end transfer command was received.	R/W
27:26	Reserved.	R/W
25:24	Packet Count M1 (PCM1) For High-Speed, High Bandwidth isochronous IN endpoints, this field in an Isoc-First TRB represents the total number of packets in the Buffer Descriptor minus 1.	R/W
23:0	Buffer Size (BUFSIZ) If CHN=0 and HWO=0 for the TRB, this field represents the total remaining Buffer Descriptor buffer size in bytes. Valid Range: 0 bytes to (16 MB - 1 byte). The hardware decrements this field to represent the remaining buffer size after data is transferred. For a Link TRB, the buffer size should be "0".	R/W
<b>DW 0F-0C</b>		
31:30	Reserved.	R/W
29:14	Stream ID / SOF Number	R/W

*Table continues on the next page...*

## Functional Description

**Table 36-8. Device Descriptor Structure Field Definitions (continued)**

Field	Description	Hardware Access
	<p>For stream-based bulk endpoints: The Stream ID of the transfer this TRB is associated with. Stream ID must be the same in all TRBs (R/W).</p> <p>For isochronous endpoints: The (micro)frame number in which the last packet of this TRB's buffer was transmitted or received (debug purposes only) (RO).</p>	
13:12	Reserved.	R/W
11	<p>Interrupt on Complete (IOC)</p> <p>When IOC is set in a TRB, and once the transfer for this buffer is completed, the core will issue XferInProgress event with IOC bit set in the event's status. This indicates the buffer is available for software to reuse or release.</p>	R
10	<p>Interrupt on Short Packet / Interrupt on Missed ISOC (ISP/IMI)</p> <p>Applicable to OUT endpoints when a short packet is received, and CSP=1 and LST=0. If this bit is 1, the core generates an XferInProgress event.</p> <p>For Isochronous endpoints: If this bit is 1, the core generates an XferInProgress event when the interval represented by the Buffer Descriptor completes with a "Missed Isoc" status.</p>	R
9:4	<p>TRB Control (TRBCTL) Indicates the type of TRB:</p> <ul style="list-style-type: none"> <li>1: Normal (Control-Data-2+ / Bulk / Interrupt) - Set TRBCTL to 1 for all TRBs used in data stage except the first TRB</li> <li>2: Control-Setup</li> <li>3: Control-Status-2 - Set TRBCTL to 3 for a SETUP request without data stage</li> <li>4: Control-Status-3 - Set TRBCTL to 4 for a SETUP request with data stage</li> <li>5: Control-Data - Set TRBCTL to 5 for the first TRB of a data stage</li> <li>6: Isochronous-First - Set TRBCTL to 6 for the first TRB of a Service Interval</li> <li>7: Isochronous</li> <li>8: Link TRB</li> <li>Others: Reserved</li> </ul>	R
3	<p>Continue on Short Packet (CSP)</p> <p>Applicable to OUT endpoints only when a short packet is received.</p> <p>If this bit is 1, the core will continue to the next Buffer Descriptor. This setting is required for isochronous endpoints.</p> <p>If this bit is 0, the core will generate an XferComplete event and remove the stream.</p>	R
2	<p>Chain Buffers (CHN)</p> <p>Applicable to IN and OUT endpoints.</p> <p>Set to 1 by software to associate this TRB with the next TRB. A Buffer Descriptor is defined as one or more TRBs. The CHN bit is used to identify the TRBs that comprise a Buffer Descriptor. The CHN bit is always 0 in the last TRB of a Buffer Descriptor and when the LST field is set to 1.</p>	R
1	<p>Last TRB (LST)</p> <p>Indicates this is the last TRB in a list. After completing the transfer for the associated buffer the core will stop the transfer for the endpoint / bulk-stream and issues an XferComplete event. The stream is automatically removed by the hardware.</p>	R
0	Hardware Owner of Descriptor (HWO) Indicates that hardware owns the TRB.	R/W

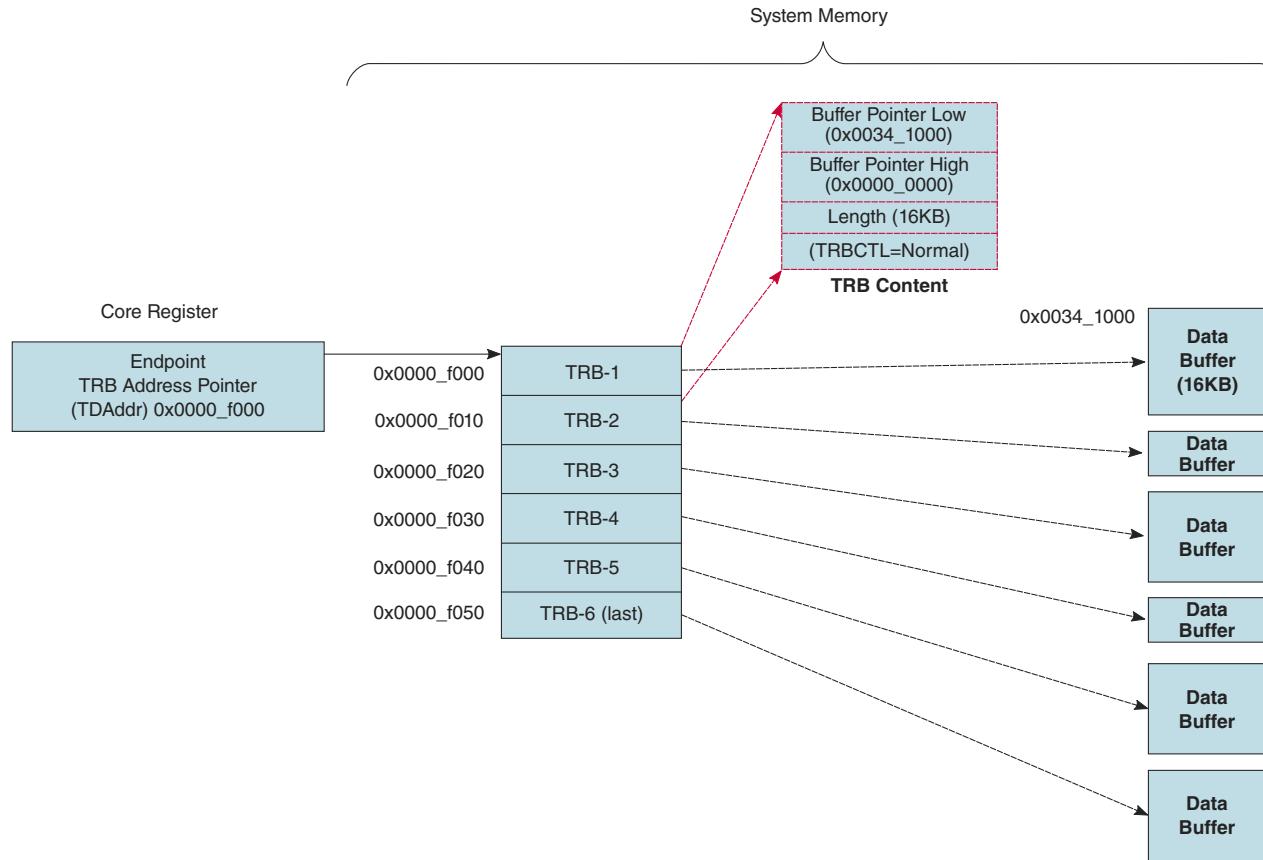
**Table 36-8. Device Descriptor Structure Field Definitions**

Field	Description	Hardware Access
	<p>Software sets this bit to 1 when it creates the TRB, and cannot modify it until hardware resets this bit to 0. However, there are exceptions for short packets on OUT endpoints and Link TRBs.</p> <p>Because the hardware autonomously checks this bit to determine if the entire TRB is valid, software must set this field to '1' after preparing the other three DWORDs of the TRB with valid information.</p>	

### 36.4.2.1.1 Normal (Control-Data/Bulk/Interrupt), Isochronous, and Status Transfer Request Block Structure

The Normal TRB is used for Bulk/Control-Data/Interrupt endpoint transfers. The data buffers can be scattered anywhere and each may have different sizes. The TRB Buffer Pointer and the Buffer Size fields point to buffer address and size respectively. The Stream ID for bulk endpoints will be programmed by the application.

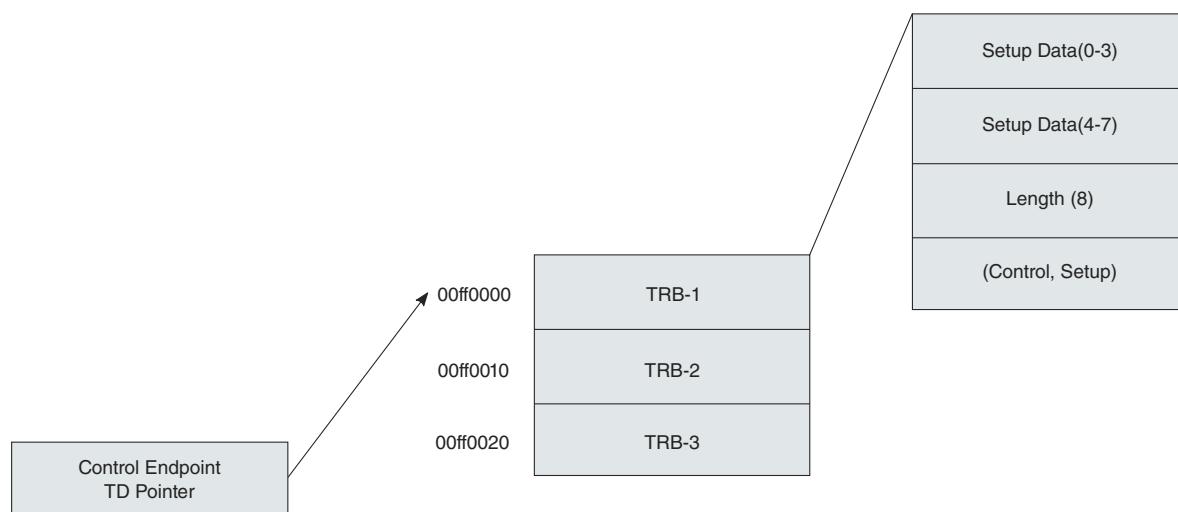
For isochronous endpoints, the first TRB in a service interval must have the Isoc-First type, the last TRB in a service interval must have CHN=0, and any other TRBs have CHN=1. The starting (micro)frame time is communicated via the Start Transfer command, and the core tracks the (micro)frame times of the subsequent Buffer Descriptors.



**Figure 36-6. Normal (Control-Data/Bulk/Interrupt) Descriptor Structure**

### 36.4.2.1.2 Setup and Status TRB Structure

To receive a SETUP packet, the driver queues up a single Setup TRB, whose buffer pointer value may be set to any address, including the address of the TRB. The buffer size must be set to 8. The core will write the 8 bytes of the received SETUP to the address requested. If the address of the TRB is used, there is no need for a separate data buffer to receive a SETUP packet. After completing Setup stage, driver will schedule Data stage and Status stage transfers. For more information, see [Control transfer programming model](#)



**Figure 36-7. SETUP Descriptor Structure with Buffer Pointing to Setup TRB**

After interpreting the SETUP bytes, software will determine if the next stage of the control transfer is a data stage or status stage.

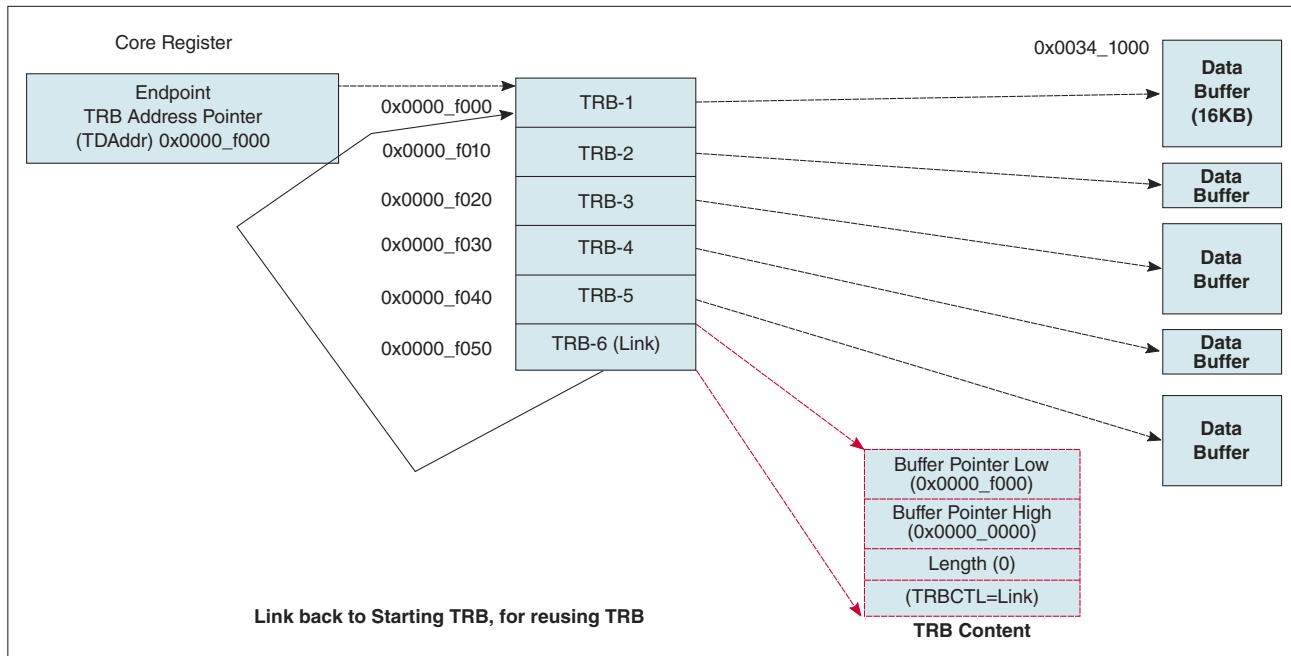
If the SETUP bytes require a 3-stage control transfer, the TRB type used in the Data stage must be Control- Data for the first TRB of the Buffer Descriptor. When the host moves on to the Status stage, the TRB Type must be Control-Status-3.

If the SETUP bytes require a 2-stage control transfer, the TRB Type must be Control- Status-2.

The Status TRB is a zero-length TRB, with an unspecified Buffer Pointer value and a Buffer Size of zero. There is no data buffer associated with a Status TRB.

### 36.4.2.1.3 Link TRB Structure

The Link TRB is used to link back to the starting TRB for reusing TRBs in a circular fashion.



**Figure 36-8. Link TRB Structure**

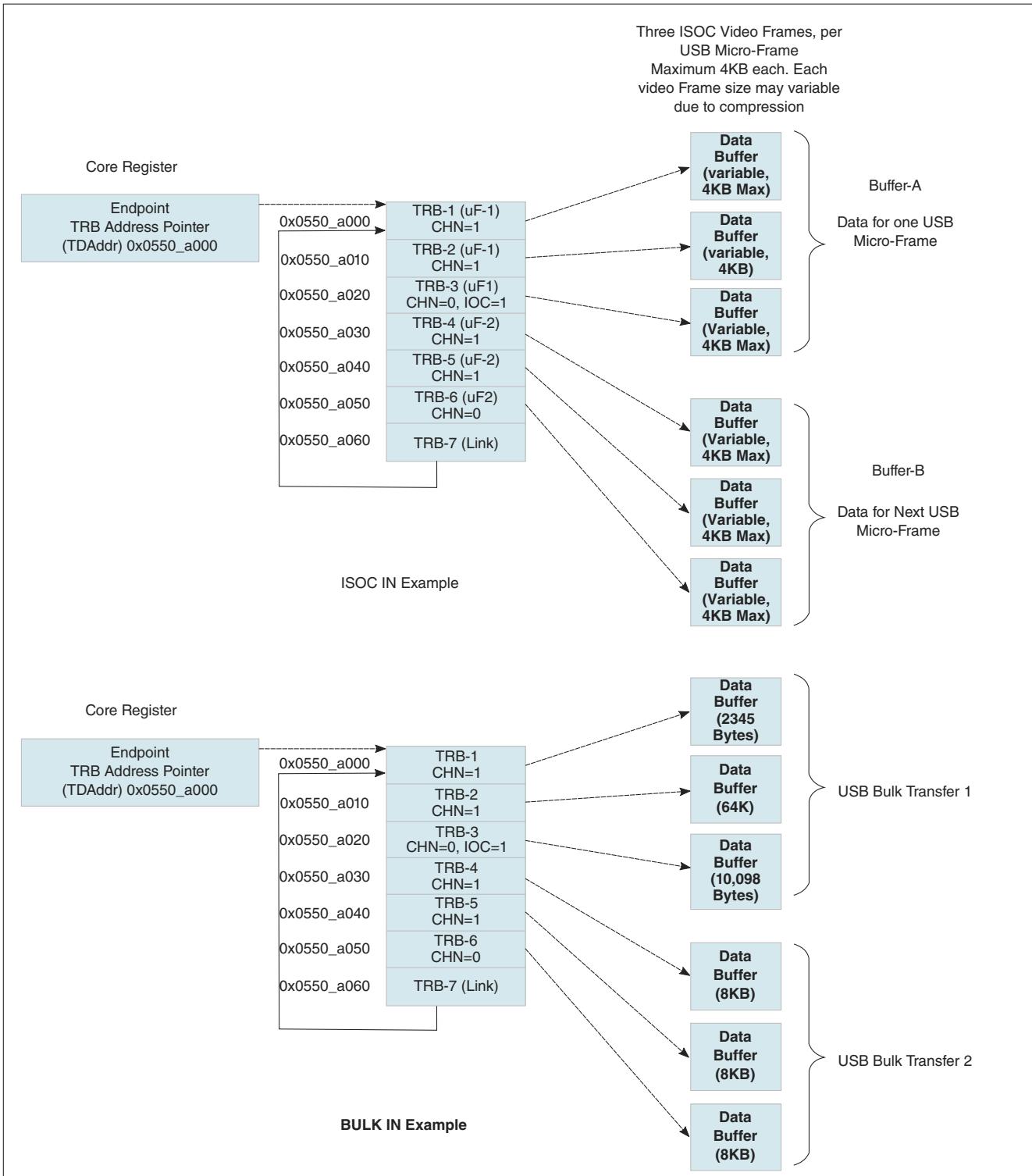
If software prepares a circular TRB list which is shorter than the number of cached TRBs (four), the core automatically detects the loop and will not re-fetch a TRB that has already been fetched. For example, for four cached TRBs the software setup three TRBs in the following way:

- TRB-1: Normal, HWO=1
- TRB-2: Normal, HWO=1
- TRB-3: Link to TRB 1

The core will fetch TRBs 1, 2, and 3. Then the core will follow the link to TRB-1, note that its address is the same as the TRB-1 that has already been fetched into the cache, and will temporarily stop fetching TRBs. When TRB-1 completes due to traffic on the USB, the core will write TRB-1 back to memory with the HWO field set to '0', generate a XferInProgress event if necessary, and will automatically attempt to fetch TRB-1 again. In most cases, the core will see the HWO field set to '0' and will stop fetching until software updates the TRB, sets the HWO field back to '1', and issues an Update Transfer command.

#### 36.4.2.1.4 Chaining Buffers (CHN) and Interrupt On Completion (IOC) Usage

This figure shows a chaining buffer example for an isochronous IN and a bulk IN transaction.

**Figure 36-9. Chaining Buffers for Isochronous and Bulk IN**

In the isochronous IN application shown in above figure, there are three video frames to be sent to the host in each microframe. Each video buffer size is maximum of 4 KB and the size may vary for each microframe depending on the compression. The CHN bit in

the TRB-3 is 0, which indicates this is the last buffer of a transfer for the given microframe. The device schedules TRB-1, 2, and 3 in the first bInterval. Similarly, TRB-6 indicates (CHN=0) microframe boundary, and TRB-4, 5, and 6 will be sent during the next bInterval.

For the last packet of last transfer in a microframe, the device internally sets last packet flag when responding at SuperSpeed for isochronous IN transfers.

The application can also use the IOC bit to receive an interrupt when the Buffer-A transfer is completed, so it can use buffer-A to send data on the bInterval following the next bInterval. As long as the driver can service the interrupt and set up data before another bInterval, the USB transfers can continue without interruption. Depending upon the system interrupt latency, you can adjust the buffer size.

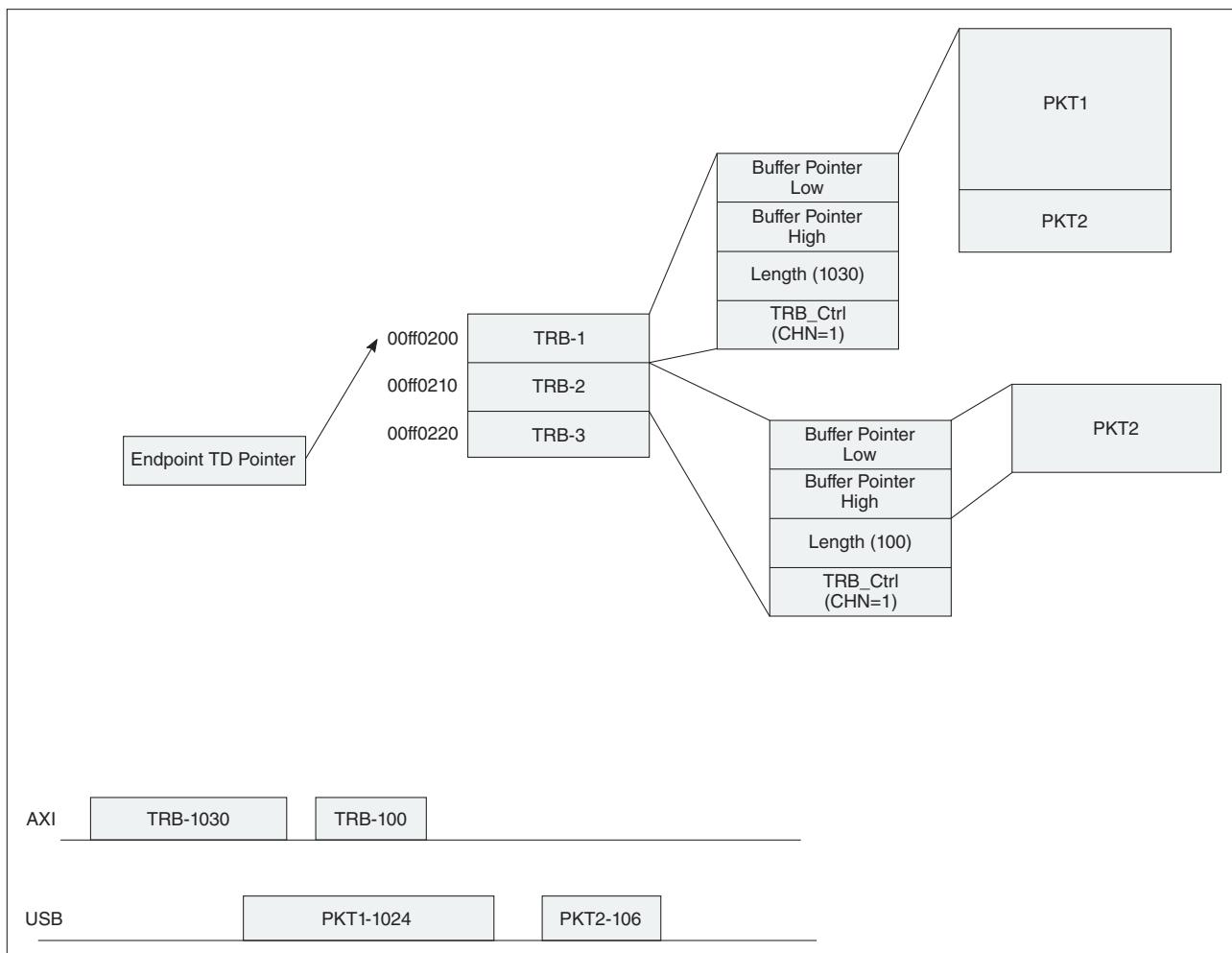
If interrupt latency is high occasionally, the software may not have time to set up the linked TRB before the next bInterval. In this case, the core, on seeing HWO bit set to 0 in the TRB, stops processing further TRBs of this endpoint. When hardware receives the Update Transfer command from software, it will re-fetch the TRB. In the case of bulk application, the core will issue the NRDY signal. In isochronous transfers, zero-length packets are sent to the host until software enables the transfer again.

In the bulk IN example, the CHN bit indicates USB Transfer boundary. During a TRB transfer, CHN indicates whether to send the bytes from next TRB buffer as part of the current transfer. For example, even though TRB-1 has 2,345 bytes, the last 297 ( $2345 - 2 * 1024 = 297$ ) bytes will be combined with the data in TRB-2 and sent as a 1,024 byte packet on the USB since CHN=1. On the other hand, when there is a short packet left in TRB-3, the short packet will be sent separately, since CHN=0.

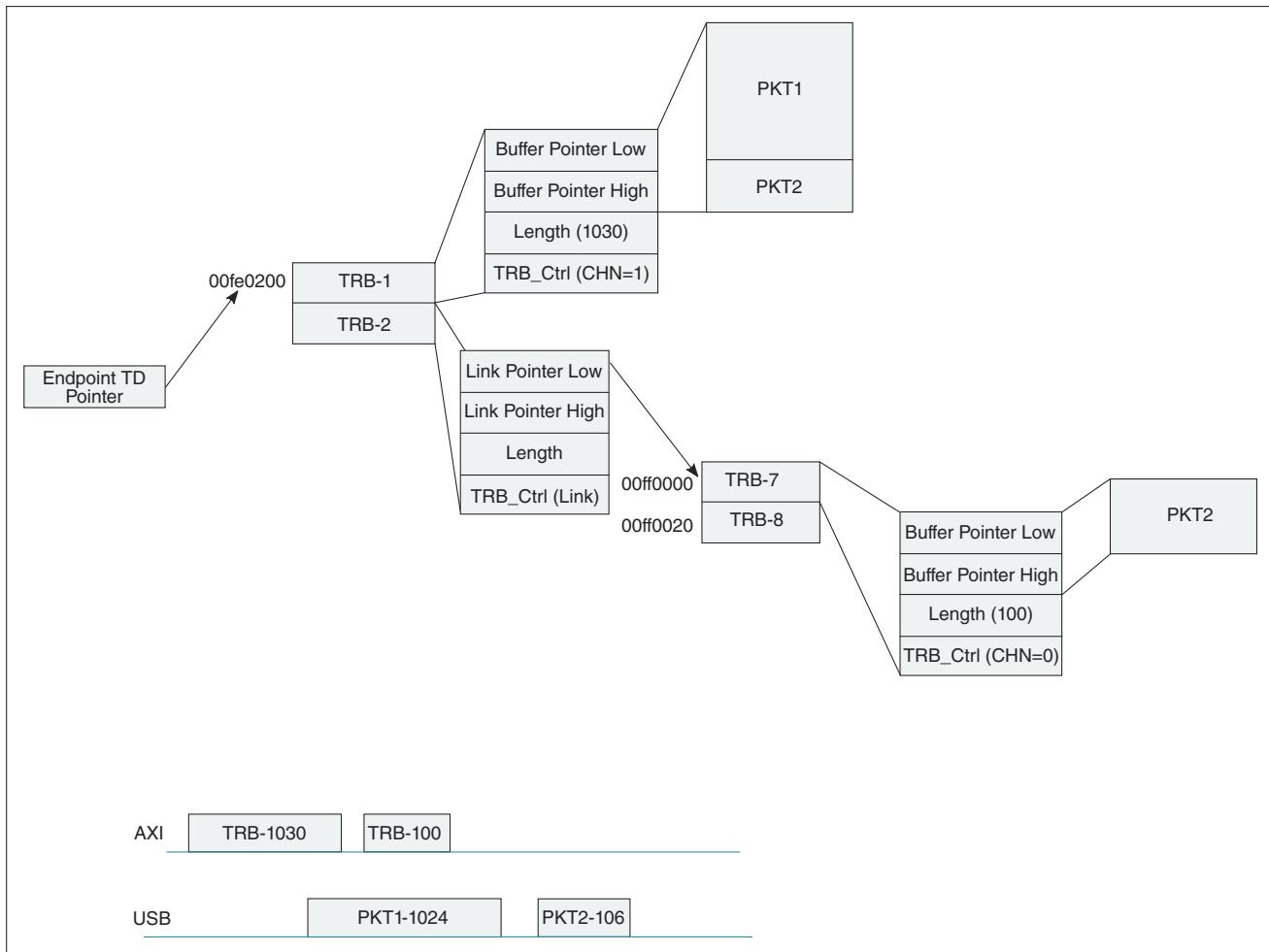
The device writes data into TRB-1 and TRB-2 for the first transfer. The CHN bit (CHN=0) in TRB-2 indicates that this is the last buffer of the transfer, then TRB-3 and TRB-4 are written for the second transfer. Similarly, the CHN bit (CHN=0) in TRB-4 indicates the second transfer boundary.

If an interrupt latency is high, software may not have time to set up the TRB. In this case, the core, on seeing the HWO bit set to 0 in the TRB, will stop processing further TRBs for the endpoint. Hardware will re-fetch the TRB when software issues the Update Transfer command. In the case of bulk, control, or interrupt endpoints, the core will issue NRDY. In the case of isochronous endpoints, packets will be dropped until software enables the transfer again.

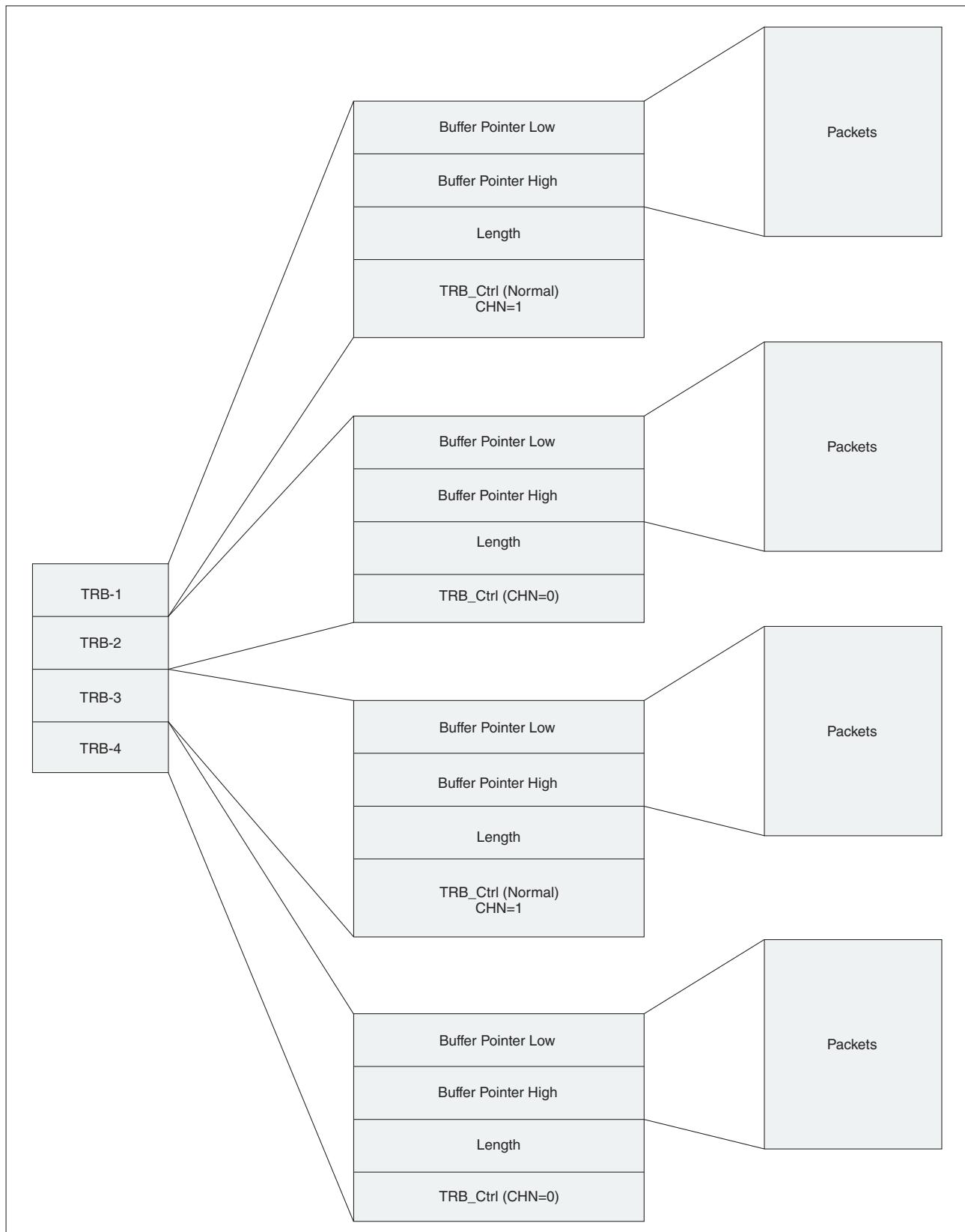
In the bulk OUT example below, when a short packet is received for TRB-1, TRB-3 will be used for the next OUT transfer and TRB-2 will be closed.



**Figure 36-10. Chaining Buffers, Example 1**



**Figure 36-11. Chaining Buffers, Example 2**



**Figure 36-12. Bulk OUT with Two Transfers**

### 36.4.2.1.5 Interrupt on Short Packet (ISP) and Continue on Short Packet (CSP) Usage

These two bits are used to schedule single or multiple OUT transfer. In most applications where only one OUT transfer per endpoint is scheduled, the software always sets ISP=1 and CSP=0. If the device receives a short packet, then it indicates a USB transfer completion through XferComplete events.

In an Ethernet-over-USB application, where software knows it is going to receive short packets, it can set up multiple transfer size buffers in one step by setting CSP=1 in all the TRBs. On a short packet, the device will update the byte count and move to next TRB. Software can also set ISP once in n number of TRBs to receive an XferInProgress event so it can process the previous short packets.

The following example shows software setting multiple 1500-byte Ethernet transfers and enabling the XferInProgress event once for four Ethernet packets to reduce the number of interrupts.

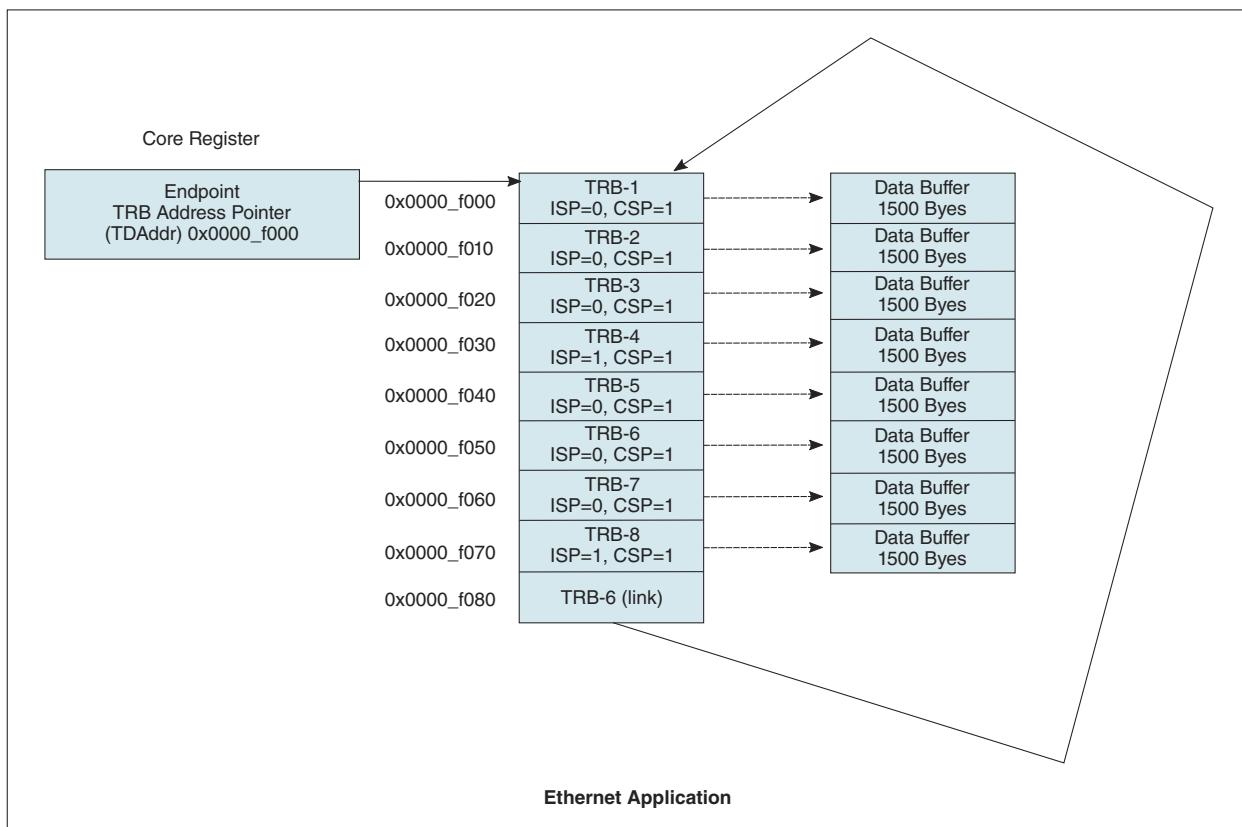
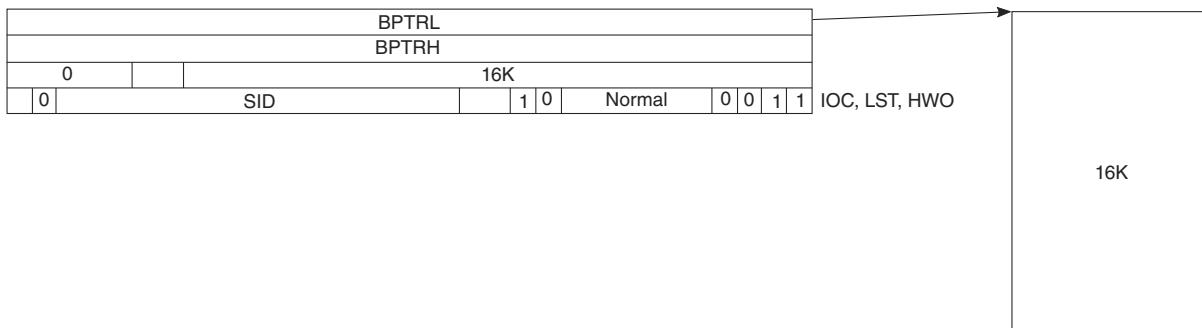


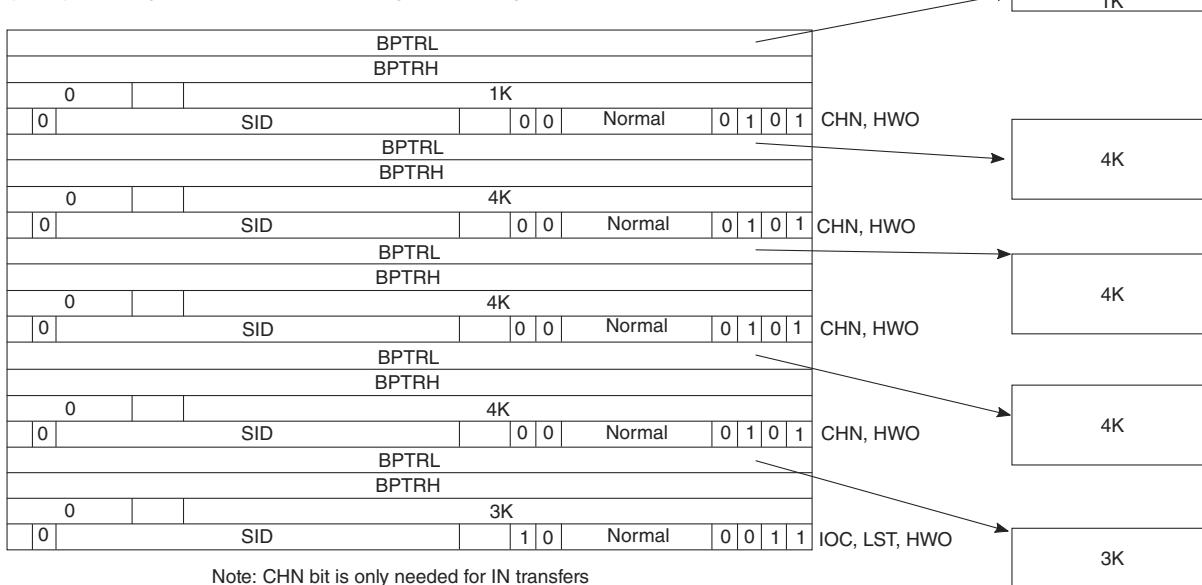
Figure 36-13. ISP and CSP Usage

### 36.4.2.1.6 Example of Setting Up TRBs

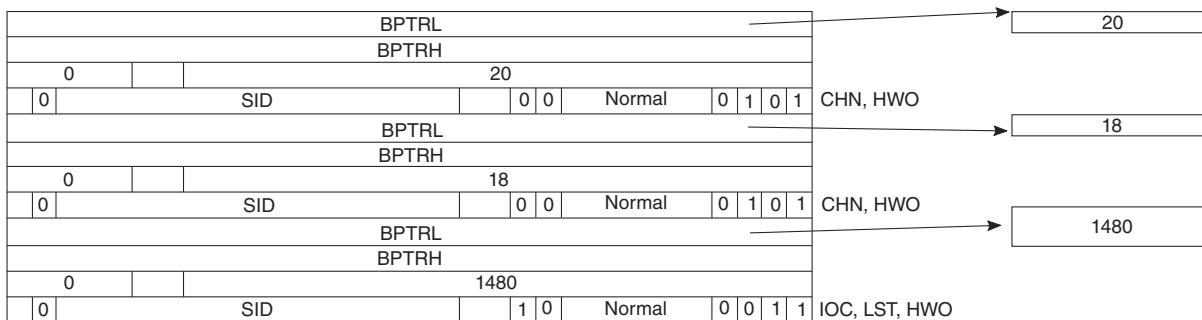
Physically Contiguous 16KB Bulk Transfer



Physically Discontiguous 16KB Bulk Transfer (e.g. Mass Storage)



Physically Discontiguous 1518 Byte Bulk Transfer (e.g. Ethernet)

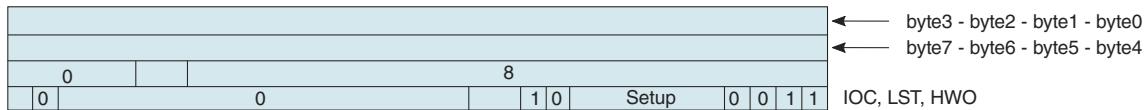


Note: CHN bit is only needed for IN transfers

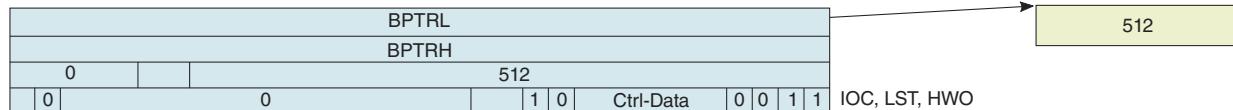
**Figure 36-14. Bulk IN TRB Examples**

## Functional Description

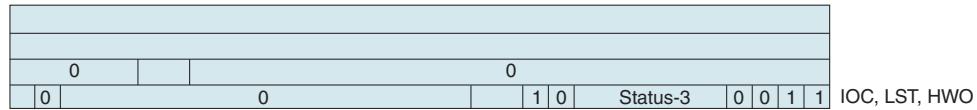
Control Write Transfer, Setup Stage



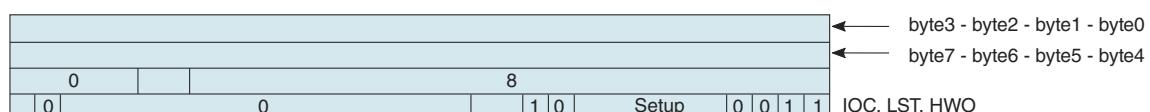
Control Write Transfer, Data Stage (Optional)



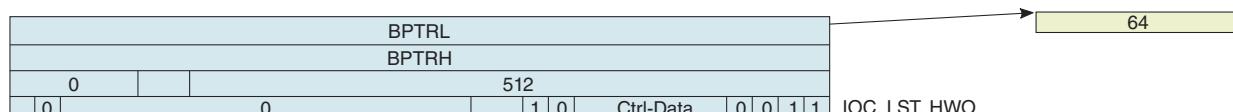
Control Write Transfer, Status Stage



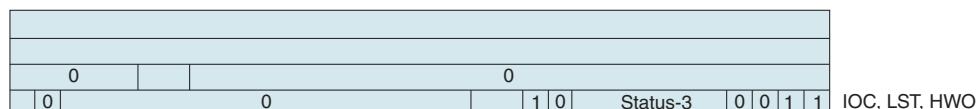
Control Read Transfer, Setup Stage



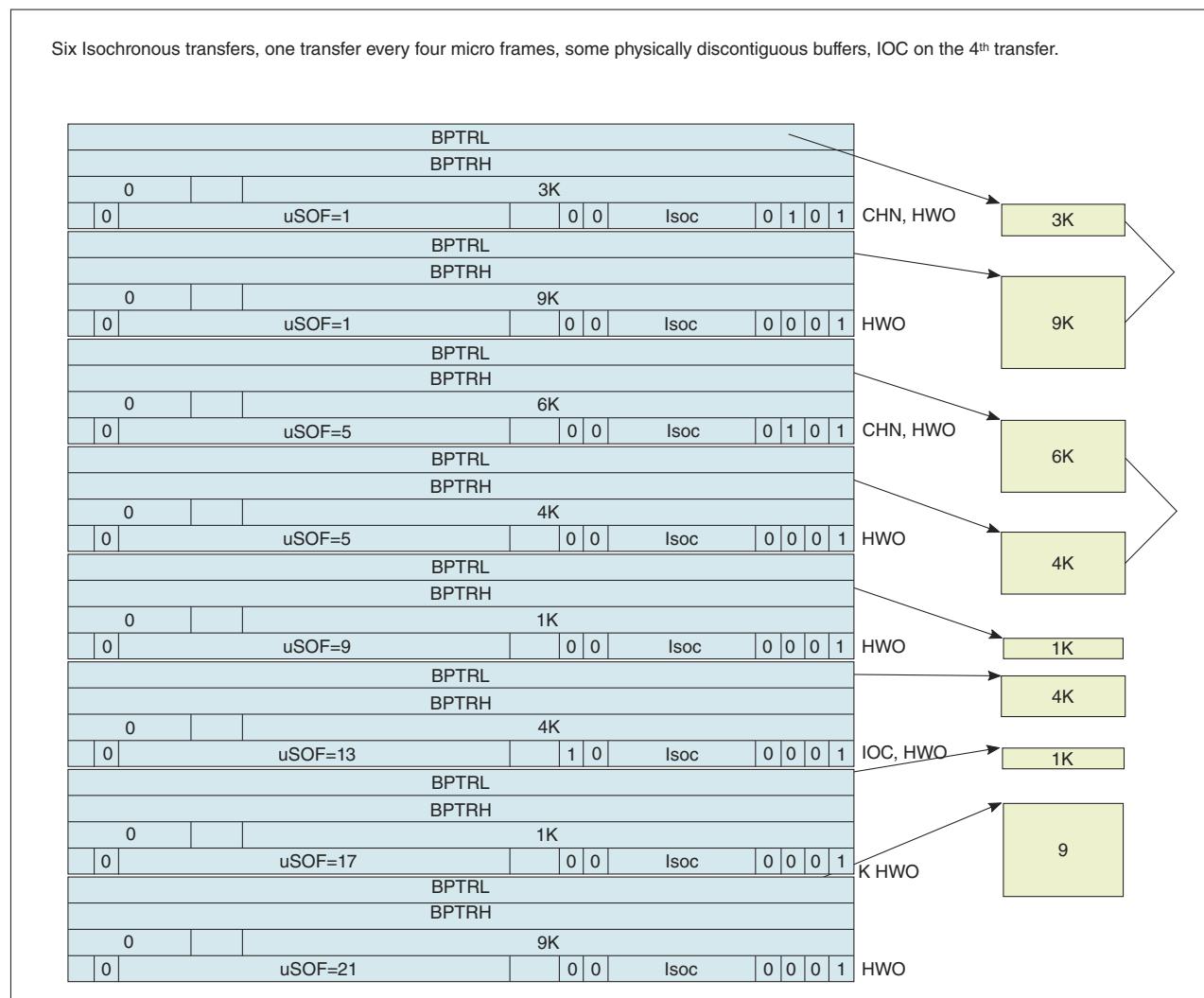
Control Read Transfer, 64B Data Stage



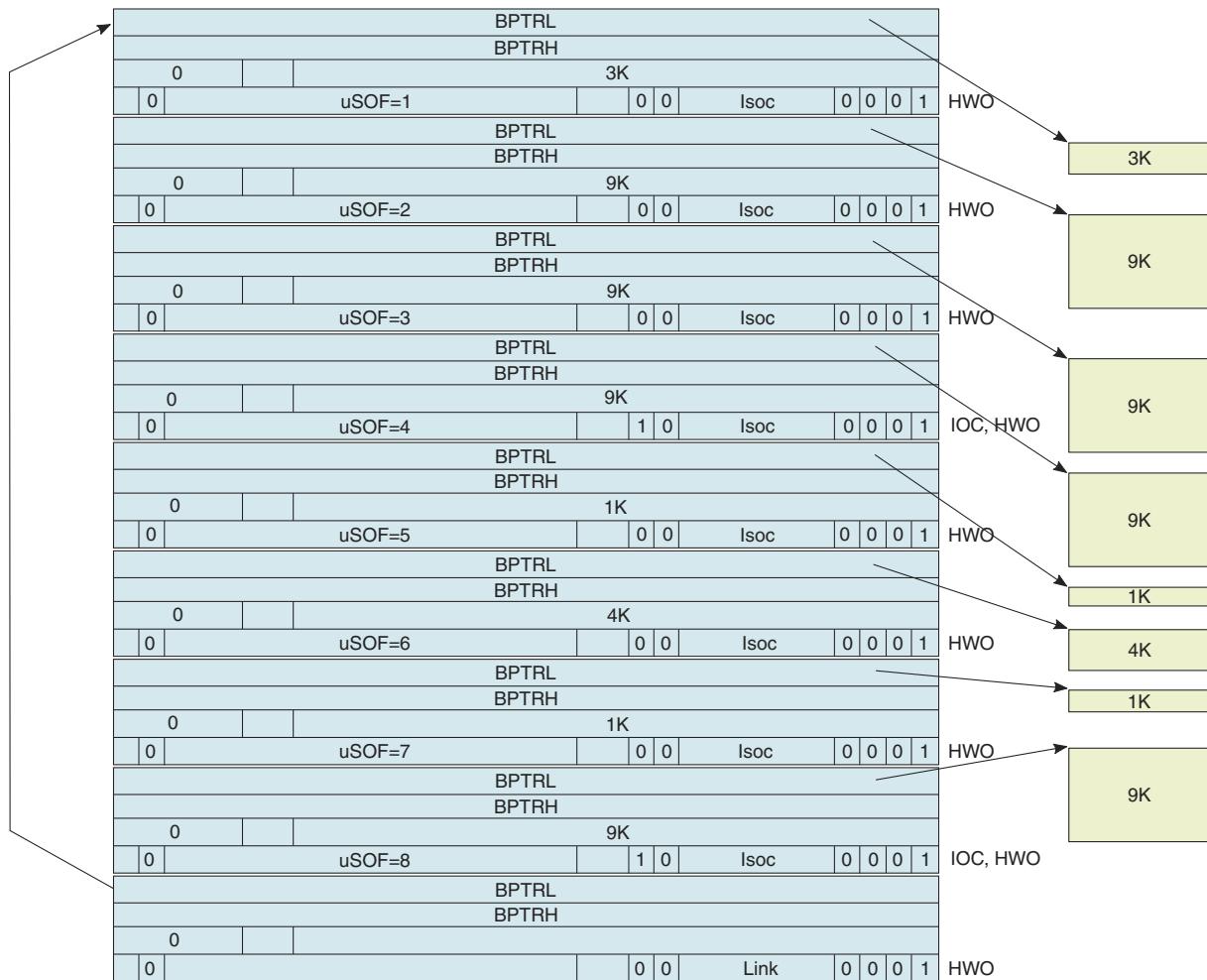
Control Read Transfer, Status Stage



**Figure 36-15. Setup/Control/Status TRB Examples**

**Figure 36-16. Isochronous IN TRB, Example 2**

Eight Isochronous Transfers, Physically Contiguous Buffers, IOC on the 4<sup>th</sup> and 8<sup>th</sup> Transfers, Circular Linked



**Figure 36-17. Isochronous IN TRB, Example 1**

### 36.4.3 Device Programming Model

#### 36.4.3.1 Register Initialization

The USB3 core contains global registers (prefixed by G\*) and device registers (prefixed by D\*) that are programmed to start operation and handle certain events. This section describes which registers must be accessed depending on the event that software is attempting to handle:

- Power-On or Soft Reset
- USB Reset Event

- Connect Done Event
- SetAddress Device Request
- SetConfiguration Device Request
- Disconnect Event
- Device-Initiated Disconnect and Reconnect

### 36.4.3.1.1 Device Power-On or Soft Reset

This section explains device core initialization after power-on or soft reset. The application must follow this initialization sequence for device mode operation.

When the core is first powered on, software initializes the following registers. The order of operations is not important, except for the first and last steps (DCTL[CSftRst]=1 and DCTL[RunStop]=1).

**Table 36-9. Power-On or Soft Reset Register Initialization**

Register	Description
DCTL	Set the CSftRst field to '1' and wait for a read to return '0'. This resets the device core.
GSBUSCFG0/1	Leave the default values, refer <a href="#">Global SoC Bus Configuration Register 0 (USB_GSBUSCFG0)</a> and <a href="#">Global SoC Bus Configuration Register 1 (USB_GSBUSCFG1)</a>
GTXTHRCFG/ GRXTHRCFG	This is required only if threshold is enabled.
GUSB2PHYCFG	Program PHY as per <a href="#">Global USB2 PHY Configuration Register (USB_GUSB2PHYCFG)</a> or leave the default values if the correct power-on values were selected.  Note: The PHY must not be enabled for auto-resume in device mode.
GUSB3PIPECTL	Program the following PHY [DatWidth] or leave the default values if the correct power-on values were selected.
GTXFIFOSIZn	Write these registers to allocate prefetch buffers for each Tx endpoint. Unless the packet sizes of the endpoints are application-specific, it is recommended to use the default value.
GRXFIFOSIZ0	Write this register to allocate the receive buffer for all endpoints. Unless the packet sizes of the endpoints are application-specific, it is recommended to use the default value.
GEVNTADR/ GEVNTSIZ/ GEVNTCOUNT	Depending on the number of interrupts allocated, program the Event Buffer Address and Size registers to point to the Event Buffer locations in system memory, the sizes of the buffers, and unmask the interrupt.  Note: USB operation stops if the Event Buffer memory is insufficient, because the core stops receiving/transmitting packets.
GCTL	Program this register to override scaledown, RAM clock select, and clock gating parameters.
DCFG	Program device speed and periodic frame interval
DEVTEN	At a minimum, enable USB Reset, Connection Done, and USB/Link State Change events
DEPCMD0	Issue a DEPSTARTCFG command with DEPEVT.XferRscldx set to 0 and CmdIOC set to 0 to initialize the transfer resource allocation. Poll CmdAct for completion.
DEPCMD0/ DEPCMD1	Issue a DEPCFG command for physical endpoints 0 & 1 with the following characteristics, and poll CmdAct for completions:  USB Endpoint Number = 0 or 1 (for physical endpoint 0 or 1) FIFONum= 0

*Table continues on the next page...*

**Table 36-9. Power-On or Soft Reset Register Initialization (continued)**

Register	Description
	XferNRdyEn and XferCmplEn = 1 Maximum Packet Size = 512 Burst Size = 0 EPTType = 2'b00 (Control)
DEPCMD0/ DEPCMD1	Issue a DEPXFERCFG command for physical endpoints 0 & 1 with DEPCMDPAR0_0/1 set to 1, and poll CmdAct for completions
DEPCMD0	Prepare a buffer for a setup packet, initialize a setup TRB, and issue a DEPSTRTRXFER command for physical endpoint 0, pointing to the setup TRB. Poll CmdAct for completion.  Note: The core attempts to fetch the setup TRB via the master interface after this command completes.
DALEPENA	Enable physical endpoints 0 & 1 by writing 0x3 to this register
DCTL	Set DCTL[RunStop] to '1' to allow the device to attach to the host. At this point, the device is ready to receive SOF packets, respond to control transfers on control endpoint 0, and generate events.

After the controller has been started, wait for the following events:

- Wait for a DEVT.USBReset event. This indicates that a reset is detected on the USB.
- Wait for a DEVT.ConnectionDone event. This event indicates the end of the reset on the USB. On this event, read the DSTS register to get the connection speed.

#### 36.4.3.1.2 Initialization on USB reset

To initialize the core as a device, during USB Reset, the application must perform the following steps:

**Table 36-10. Initialization on USB reset**

Register	Description
DEPCMD0	If a control transfer is still in progress, complete it and get the core into the "Setup a Control-Setup TRB / Start Transfer" state
DEPCMDn	Issue a DEPENDXFER command for any active transfers (except for the default control endpoint 0)
DEPCMDn	Issue a DEPCSTALL (ClearStall) command for any endpoint in STALL mode prior to the USB Reset (excluding control endpoints)
DCFG	Set DevAddr to '0'

#### NOTE

Special Reset Considerations for Default Control Endpoint 0:

The default control endpoint is not affected by a USB Reset, so software must continue following the software flow for control transfers explained in "Control Transfer Programming Model" even across a USB Reset event. Resources can only be assigned

to the default control endpoint once after a power on or soft reset.

### NOTE

The USB PHY frequency (SCFG\_USB\_REFCLK\_SELCRn) should be programmed in the PBI phase such that the USB PHY comes out of reset before SYSTEM\_READY.

#### 36.4.3.1.3 Initialization on connect done

When this event is received, software must perform the following steps:

**Table 36-11. Initialization on connect done**

Register	Description
DSTS	Read this register to obtain the connection speed
GCTL	Program the RAMClkSel field to select the correct clock for the RAM clock domain. This field is reset to 0 after USB reset, so it must be reprogrammed each time on Connect Done.
DEPCMD0/ DEPCMD1	Issue a DEPCFG command (with Config Action set to "Modify") for physical endpoints 0 & 1 using the same endpoint characteristics from Power-On Reset, but set MaxPacketSize to 512 (SuperSpeed), 64 (High-Speed), 8/16/32/64 (Full-Speed), or 8 (Low-Speed).
GUSB2CFG/ GUSB3PIPECTL	Depending on the connected speed, write to the other PHY's control register to suspend it
GTXFIFOSIZn	(optional) Based on the new MaxPacketSize of IN endpoint 0, software may choose to re-allocate the TX FIFO sizes by writing to these registers.

#### 36.4.3.1.4 Initialization on SetAddress request

When the application receives a SetAddress request in a SETUP packet, it performs the following steps:

**Table 36-12. Initialization on SetAddress request**

Register	Description
DCFG	Program the DCFG register with the device address received as part of the SetAddress request when SETUP packet is decoded.
DEPCMD1	After receiving the XferNotReady(Status) event, acknowledge the status stage by issuing a DEPSTRTXFER command pointing to a Status TRB. This step must be done after the DCFG register is programmed with the new device address.

At this point, the device is ready to receive micro-SOF/ITP and is configured to receive control transfers on control endpoint 0 with a new address assigned.

### 36.4.3.1.5 Initialization on SetConfiguration or SetInterface Request

When the application receives a SetConfiguration or SetInterface request in a SETUP packet, it performs the following steps:

**Table 36-13. Initialization on SetConfiguration or SetInterface Request**

Register	Description
DALEPENA	Set this register to 0x3 to disable all endpoints other than the default control endpoint 0.
DEPCMDn	Issue a DEPENDXFER command for any active transfers (except for the default control endpoint 0).
DEPCMD1	Issue a DEPCFG command (with Config Action field set to "Modify") for physical endpoint 1 using the current endpoint characteristics to re-initialize the TX FIFO allocation.
DEPCMD0	Issue a DEPSTARTCFG command with DEPCMD0.XferRscldx set to 2 to re-initialize the transfer resource allocation.
DEPCMDn	Issue a DEPCFG command (with the Config Action field set to "Initialize") for each endpoint that is present in the new configuration (except for the default control endpoint).  Note: Control endpoints are bi-directional and must use consecutive even/odd physical endpoint numbers for the OUT/IN direction (such as 2/3, or 4/5), and the FIFONum must be configured to the same value in both directions.
DEPCMDn	Issue a DEPXFERCFG command for each endpoint that is present in the new configuration (except for the default control endpoint 0).
GTXFIFOSIZn	(optional) Based on the new configuration of IN endpoints, software may choose to re-allocate the TX FIFO sizes by writing to these registers.
DALEPENA	Enable the logical endpoints that are active in the new configuration.
DEPCMD1	After receiving the XferNotReady(Status) event, acknowledge the status stage by issuing a DEPSTRTRXFER command pointing to a Status TRB. This step must be done after the previous steps to ensure the host does not access any other endpoints that are being set up.

At this point, the core is prepared to accept Start Transfer commands for the newly-configured endpoints. For information on how to set up transfers, see the [Operational Model](#).

### 36.4.3.1.6 Alternate Initialization on SetInterface Request

When handling a SetInterface device request, another possibility is that software may reconfigure existing endpoints instead of starting the configuration from the beginning. This is only possible if no TX FIFOs need to be reassigned. This flow also assumes that the endpoints that are being removed or reconfigured in the new interface have halted their traffic prior to the host issuing the SetInterface request.

**Table 36-14. Alternate Initialization on SetInterface Request**

Register	Description
DEPCMDn	Make sure there are no transfers still active on the endpoints that are changing in the new interface. This is normally ensured by the host prior to issuing the SetInterface, but if not, issue an End Transfer command for the transfer on each endpoint that is changing.

*Table continues on the next page...*

**Table 36-14. Alternate Initialization on SetInterface Request (continued)**

DEPCMDn	Issue a DEPCFG command (with the Config Action field set to "Initialize") for each endpoint that is changing in the new configuration. The side effect of the DEPCFG command is that the endpoint's sequence number is automatically set to 0.
DALEPENA	Write this register with all the endpoints that are enabled (including the ones that are not changing).
DEPCMDn	Using the StartXfer command, acknowledge the Status stage response for the SetInterface request.

### 36.4.3.1.7 Initialization on Disconnect Event

When the application receives a Disconnect event, it must set DCTL[8:5] to 5. Other than this, the core does not require any initialization. Because the DCTL[RunStop] bit is still '1', the device attempts to reconnect to the host, at which time a USB Reset and Connect Done event occurs. However, if the application does not want to attempt to reconnect to the host, it should perform the steps in the next section.

#### NOTE

When DCFG[DevSpd] is programmed for 2.0 only mode (such as, High-Speed or Full-Speed), and if the application wants to issue any commands to clear any pending transfers during a Disconnect interrupt, then it has to disable `USB_GUSB2PHYCFG[SUSPENDUSB20]` before issuing any commands and re-enable it after the commands have completed.

### 36.4.3.1.8 Device-initiated disconnect

If the application wants to disconnect from the host, it should perform the following actions:

**Table 36-15. Initialization on device-initiated disconnect**

Register	Description
DEPCMD0	If a control transfer is still in progress, complete it and get the core into the "Setup a Control-Setup TRB / Start Transfer" state
DEPCMDn	Issue a DEPENDXFER command for any active transfers (except for the default control endpoint 0)
DCTL	Set DCTL[RunStop] to '0' to disconnect from the host
DSTS	Poll [DEVCTRLHLT] until it is '1'

At this point, the device is disconnected from the host and will not attempt to reconnect.

### 36.4.3.1.9 Reconnect after Device-Initiated Disconnect

If the application decides to reconnect to the host, it must follow the steps in [Device Power-On or Soft Reset](#).

### 36.4.3.2 Operational Model

The following sections describes the processes and data structures that the core uses to implement the USB 3.0 specification.

#### 36.4.3.2.1 USB and Physical Endpoints

Endpoints are referred to in two ways:

- As a USB endpoint number: USB endpoints are defined in the USB specification.
- As a physical endpoint resource number: The hardware has a fixed number of physical endpoint resources. Each resource is unidirectional and can be configured to refer to either direction of any USB endpoint.

The software always works on physical endpoints and it knows how physical endpoint corresponds to USB endpoints. DALEPENA is the only register that has one enable bit per USB endpoint.

During SetConfiguration, software maps physical endpoint resources to the required USB endpoints. When a USB request comes in, the USB endpoint number gets converted to a physical endpoint number in the core. Similarly, when a USB packet is sent out, the physical endpoint number is converted to the USB endpoint and sent out.

A USB control endpoint requires two physical endpoints. One physical endpoint is mapped to the OUT direction of the Control endpoint, and the other one is mapped to the IN direction of the Control endpoint. Specifically the two physical endpoints are used as the following:

- The Setup stage of any control transfer uses the OUT direction physical endpoint.
- For a control write or 2-stage transfer, the Data stage (if present) uses the OUT direction physical endpoint and the Status stage uses the IN direction physical endpoint.
- For a control read transfer, the Data stage uses the IN direction physical endpoint and the Status stage uses the OUT direction physical endpoint.

### 36.4.3.2.2 Event Buffers

Hardware passes command completion, transfer progress, and asynchronous events to software through one or more Event Buffers.

To configure an Event Buffer, the software performs the following steps:

1. Sets up an empty buffer in system memory.
2. Writes the address of the beginning of the buffer into GEVNTADR. This address must be aligned to the Event Buffer size.
3. Writes the size of the buffer and interrupt mask into GEVNTSIZ. Depending on your system interrupt latency, enough Event Buffer space must be allocated to avoid lost interrupts or reduced performance.
4. Write a 0 into the GEVNTCOUNT register. This must be the last step, as it enables the Event Buffer. After the Event Buffer has been configured, software must not change the size or address.

There is one interrupt line per Event Buffer that indicates there are one or more events present. Software reads one or more events out of the buffer and indicates to hardware how many events it processed by writing the byte count to the GEVNTCOUNT register.

Clock crossing delays may result in the interrupt's continual assertion after software acknowledges the last event. Therefore, when the interrupt line is asserted, software must read the GEVNTCOUNT register and only process events if the GEVNTCOUNT is greater than 0.

The first event produced by the core after the Event Buffer is configured will be written to the address specified in GEVNTADR. Most events are 32 bits, and subsequent events will be written to the address (PreviousEventAddress + 4). When that address exceeds the sum of GEVNTADR and GEVNTSIZ, the core wraps around to the first GEVNTADR value. In this way, the Event Buffer operates like a circular buffer with hardware writing to the "tail" of the buffer and software reading from the "head."

Most events are exactly 4 bytes in size, but there is one exception: The Vendor Device Test LMP Received Event (VndrDevTstRcved) is a 12 byte event that includes a header in the first 4 bytes and the contents of the LMP in the following 8 bytes.

When an event occurs within the core, hardware checks the enable bit that corresponds to the event to decide whether the event will be written to the Event Buffer or not. The Event Buffer contains one of the following types of information, depending on the value of the lower bits of the event:

- Device Endpoint-Specific Event (DEPEVT) (Event[0] = 0x0)
  - The DEPCFG endpoint-specific command specifies the bits that are enabled and which Event Buffer to use for these events.
- Device-Specific Event (DEVT) (Event[0] = 0x1, Event[7:1] = 0x00)

## Functional Description

- The Generic Command Complete event is enabled through the DGCMD[CmdIOC] field when the command is issued.
- The Event Buffer Overflow Event cannot be disabled and is written to the Event Buffer that encounters the overflow.
- The rest of the Device-Specific events are enabled through the DEVTEEN register.
- Except for the Event Buffer Overflow Event, these events are written to the Event Buffer specified in the DCFG[IntrNum] field.

The core always leaves one entry free in each Event Buffer. When the Event Buffer is almost full, hardware writes the Event Buffer Overflow event and the USB will eventually get stalled when endpoints start responding NRDY or the link layer will stop returning credits (in SuperSpeed). This event is an indication to software that it is not processing events quickly enough. During this time, events will be queued up internally. When software frees up Event Buffer space, the queued up events will be written out and the USB will return to normal operation.

### Event Buffer Content for Device Endpoint-Specific Events (DEPEVT)

**Table 36-16. Device Endpoint-n Events: DEPEVT**

Field	Description
31–16	<p>Event Parameters (EventParam)</p> <p>For XferNotReady, XferComplete, and Stream events on Bulk Endpoints:</p> <p>[31-16] StreamID. Applies only to bulk endpoints that support streams. This indicates the Stream ID <sup>1</sup> of the transfer for which the event is generated</p> <p>For XferInProgress:</p> <p>[31-16] Isochronous Microframe Number (IsocMicroFrameNum). Indicates the microframe number of the beginning of the interval that generated the XferInProgress event (debug purposes only)</p> <p>For XferNotReady events on Isochronous Endpoints:</p> <p>[31-16] Isochronous Microframe Number (IsocMicroFrameNum). Indicates the microframe number during which the endpoint was not ready</p> <p><b>NOTE:</b> USB 3.0 core represents USB bus time as a 14-bit value on the bus and also in the DSTS register (DSTS[SOFFN]), but as a 16-bit value in the XferNotReady event. Use the 16-bit value to interact with Isochronous endpoints via the StartXfer command. The extra two bits that the USB 3.0 core produces will be necessary for handling wrap-around conditions in the interaction between software and hardware.</p> <p>EPCmdCmplt events</p> <p>For all EPCmdCmplt events</p> <p>[27-24]- Command Type. The command type that completed (Valid only in a DEPEVT event).</p> <p>Undefined when read from the DEPCMD.EventParam field).</p> <p>For EPCmdCmplt event in response to Start Transfer command.</p> <p>[22-16] Transfer Resource Index (XferRscldx). The internal hardware transfer resource index assigned to this transfer. This index must be used in all Update Transfer and End Transfer commands.</p>
15–12	Event Status (EventStatus)

*Table continues on the next page...*

**Table 36-16. Device Endpoint-n Events: DEPEVT (continued)**

Field	Description
	<p>Within an XferNotReady event-</p> <p>[15]: Indicates the reason why the XferNotReady event is generated:</p> <ul style="list-style-type: none"> <li>0- XferNotActive- Host initiated a transfer, but the requested transfer is not present in the hardware</li> <li>1- XferActive- Host initiated a transfer, the transfer is present, but no valid TRBs are available</li> </ul> <p>[14]: Not Used</p> <p>[13:12]: For control endpoints, indicates what stage was requested when the transfer was not ready:</p> <ul style="list-style-type: none"> <li>2'b01- Control Data Request</li> <li>2'b10- Control Status Request</li> </ul> <p>Within an XferComplete or XferInProgress event-</p> <p>[15]: LST bit of the completed TRB (XferComplete only)</p> <p>[15]: MissedIsoc: Indicates the interval did not complete successfully (XferInProgress only)</p> <p>[14]: IOC bit of the TRB that completed</p> <p>[13]: Indicates the TRB completed with a short packet reception or the last packet of an isochronous interval</p> <p>[12]: Reserved</p> <p>If the host aborts the data stage of a control transfer, software may receive a XferComplete event with the EventStatus field equal to '0'. This is a valid event that must be processed as a part of the <a href="#">Control transfer programming model</a>.</p> <p>Within a Stream Event-</p> <p>[15:12]:</p> <ul style="list-style-type: none"> <li>- 4'h2- StreamNotFound- This stream event is issued when the stream-capable endpoint performed a search in its transfer resource cache, but could not find an active and ready stream.</li> <li>- 4'h1- StreamFound- This stream event is issued when the stream-capable endpoint found an active and ready stream in its transfer resource cache, and initiated traffic for that stream to the host. The ID of the selected Stream is in the EventParam field.</li> </ul> <p>In response to a Start Transfer command-</p> <p>4'h2- Indicates expiry of the bus time reflected in the Start Transfer command.</p> <p>4'h1- Indicates there is no transfer resource available on the endpoint.</p> <p>In response to a Set Transfer Resource (DEPXFERCFG) command-</p> <p>4'h1- Indicates an error has occurred because software is requesting more transfer resources to be assigned than have been configured in the hardware.</p> <p>In response to a End Transfer command-</p> <p>4'h1- Indicates an invalid transfer resource was specified.</p>
11–10	This field is reserved.
9–6	<p>4'h7- Endpoint Command Complete (EPCmdCmplt)</p> <p>Indicates software may issue another Device Endpoint command to the endpoint.</p> <p>When issued in response to an End Transfer command, indicates that DMA stopped for the endpoint.</p> <p>For all other commands, this event does not imply that all effects of the command took place. The DEPCMD register contains the same status information present in the Event Status Bits field.</p> <p>4'h6- Stream Event (StreamEvt)</p>

*Table continues on the next page...*

**Table 36-16. Device Endpoint-n Events: DEPEVT (continued)**

Field	Description
	<p>Indicates that a stream-capable endpoint initiated a search within its transfer resource cache. The result of the search is in the EventStatus field (Found or NotFound).</p> <p>4'h5- Reserved</p> <p>4'h4 - Reserved</p> <p>4'h3- XferNotReady Event (XferNotReady)</p> <p>Indicates receipt of a transaction when no TRBs are available for the endpoint. For isochronous IN endpoints, a zero-length packet is automatically sent by hardware and NRDY for non-isochronous endpoints.</p> <p>The application must enable this event if it plans to issue Start Transfer on demand.</p> <p>This event can happen when software issues a Start Transfer or Update Transfer. In this case, software must ignore this event because it has already issued the Start Transfer or Update Transfer.</p> <p>XferNotReady is generated when the core responds NRDY to the host on the USB. It is useful in the beginning of a transfer when software wants to wait for the host to start polling the endpoint before setting up TRBs, but it is not efficient to use this event to determine when the core has run out of TRBs. For determining when the core has processed TRBs and needs software to setup more, use the IOC field in a TRB along with the XferInProgress event that is generated when the core completes a TRB with IOC=1.</p> <p>For non-stream-capable endpoints, the hardware filters multiple events if the host continues transactions, even after the core responds with NRDY. This internal filter is reset after software issues any endpoint command to this endpoint.</p> <p>For stream-capable endpoints, this event is generated each time the host attempts a transaction, even if the core responds with NRDY.</p> <p>For isochronous endpoints, this event is generated only once prior to the Start Transfer command to communicate the current bus time.</p> <p>For additional information, see the Event Status field.</p> <p>4'h2- XferInProgress Event (XferInProgress)</p> <p>Applies to IN and OUT endpoints. Indicates an EP/Stream specific event happened and it is continuing the transfer.</p> <p>For additional information, see the <a href="#">Device Programming Model</a>.</p> <p>4'h1- XferComplete Event (XferComplete)</p> <p>Applies to IN and OUT endpoints. Indicates that a EP/Stream transfer completed and the core stopped the transfer.</p> <p>For additional information, see the <a href="#">Device Programming Model</a>.</p> <p>4'h0- Reserved</p>
5-1	Physical Endpoint Number (0-31)
0	1'b0- Indicates that this is an endpoint-specific event.

1. Note that term Stream ID used in USB chapter is different from Arm SMMU terminology.

## Event Buffer Content for Device-Specific Events (DEVT)

**Table 36-17. Device-specific events: DEVT**

Field	Description
31-25	Reserved

*Table continues on the next page...*

**Table 36-17. Device-specific events: DEVT (continued)**

Field	Description
24-16	<p>Event Information Bits (EvtInfo)</p> <p>For a USB/Link State change Event, this field indicates the state of the link- EvtInfo[4] - SuperSpeed event. Set to 1 for SS; Set to 0 for non-SS.</p> <p>EvtInfo[3:0] - Link State. Indicates link state at the time of the event. Follows same encoding as DSTS link state bits.</p>
15-12	Reserved
11-8	<p>11- Event Buffer Overflow Event (EvntOverflow)</p> <p>The core writes this event to indicate there is no space in the event buffer, and one or more Device-specific events may have been dropped after this event. Endpoint-Specific events will not be dropped, but they will be delayed which can cause a drop in performance on the USB.</p> <p>Software uses this event as a warning that the Event Buffer is too small and the core should be reconfigured with a larger Event Buffer or software needs to process events more quickly. In order to avoid repeated Event Buffer Overflow Events, software must free up space in the Event Buffer by acknowledging more than 1 event (writing a value greater than 4 to the GEVNTCOUNT register). This event cannot be disabled, and is always written to the Event Buffer that encounters the overflow.</p> <p>10- Generic Command Complete Event (CmdCmplt)</p> <p>The core writes this event to indicate the generic command is complete.</p> <p>9- Erratic Error Event (ErrticErr)</p> <p>The core writes this event to report erratic errors (phy_rxvalid_i/phy_rxvldh_i or phy_rxactive_i is asserted for at least 2 ms, due to PHY error) seen on the UTMI+.</p> <p>Due to erratic errors, the USB 3.0 core goes into Suspended state and a USB/Link state change event (ULStChng) is generated to the application.</p> <p>If the early suspend is asserted due to an erratic error, the application can only perform a soft disconnect recover.</p> <p>For SuperSpeed operation, the PHY erratic error event indicates that the PIPE is not responding to the PHY command (for example, pipe3_PowerDown change or receiver detection). After the core requested any PHY command, if it did not receive pipe3_PhysStatus within 100ms, the core will timeout and generate the erratic error event. After generating the erratic error, the core assumes that the PHY command is completed successfully and proceeds with the next pending PHY command. Software must reset the controller on receiving the erratic error.</p> <p>8- Reserved</p> <p>7- Start of (micro)Frame (Sof)</p> <p>6- USB Suspend Entry Event</p> <p>This event provides a notification that the link has gone to a suspend state (L2, U3, or L1). The existing Link State Change event (3) provides the same information, but is generated for every link state change.</p> <p>5- Reserved</p> <p>4- Resume/Remote Wakeup Detected Event (WkUpEvt)</p> <p>This event is generated when the host initiates a resume condition on the USB bus. This event is not generated by the device initiating a remote wakeup to the host.</p> <p>3- USB/Link State Change (ULStChng)</p> <p>The core writes this event to indicate a change of USB or Link state. Bits 23-16 (EvtInfo) of this event indicate the Link Status.</p> <p>The event is generated in SuperSpeed when the link LTSSM state changes, except when LTSSM exits from HOT_RESET, POLL or LTSSM enters into RECOVERY.</p>

*Table continues on the next page...*

**Table 36-17. Device-specific events: DEVT (continued)**

Field	Description
	2- Connection Done (ConnectDone) 1- USB Reset (USBRst) 0- Disconnect Detected Event (DisconnEvt) Note: This event is generated only if VBUS is off.
7-1	7'h00 - Device Specific Event
0	1'b1 - Non-Endpoint-Specific Event

### 36.4.3.2.3 Transfer and Buffer Rules

Buffers that are used to transfer data to and from an endpoint are defined using a Buffer Descriptor, which consists of one or more Transfer Request Blocks (TRBs). A CHN flag in the TRB is used to identify the TRBs that comprise a Buffer Descriptor. Therefore, a Buffer Descriptor refers to a consecutive set of TRB data structures where the CHN flag is set in all TRBs, except the last. Note that a Buffer Descriptor may consist of a single TRB, whose CHN flag will not be set.

Software communicates the location of the first TRB by using the Start Transfer command on an endpoint. Even endpoints that are not stream-capable use this command. The core fetches TRBs from external memory starting at the address provided in the Start Transfer command parameters, continuing in a linear fashion and following the Link TRB until a TRB is encountered that has its LST bit set to '1' or HWO bit set to '0'. Therefore, software must ensure that it has valid TRBs prepared before issuing the Start Transfer command to prevent the core from reading uninitialized memory.

Descriptor fetch requests are buffered within the hardware and handled separately from the progress of the current transfer. Therefore, it is possible that the core completes a transfer with XferComplete but still continues reading TRBs that it has not cached yet. If software is immediately de-allocating the memory for TRBs based on the XferComplete event, it is recommended that software issue an End Transfer command for the endpoint/transfer resource prior to de-allocating the memory. The completion of the End Transfer command flushes out any pipelined descriptor fetches and avoids a potential bus error.

While processing TRBs, two conditions may cause the core to write out an event and raise an interrupt line:

- TRB Complete:

- For OUT endpoints, a packet is received which reduces the remaining byte count in the TRB buffer to zero.
- For IN endpoints, an acknowledgement is received for a transmitted packet which reduces the remaining byte count in the TRB buffer to zero.
- Short Packet Received:
  - For OUT endpoints only. While writing to a TRB buffer, the endpoint receives a packet that is smaller than the endpoint's MaxPacketSize.

This table shows the USB 3.0 Core Actions Based on TRB Control Bits.

**Table 36-18. USB 3.0 Core Actions Based on TRB Control Bits**

Direction	TRB Complete Packet	ISP	IOC	CHN	LST	CSP	Action
IN	Yes	-	X	0	1	-	XferComplete event
IN	Yes	-	0	X	0	-	No event
IN	Yes	-	1	X	0	-	XferInProgress event
OUT	Yes No	-	X	0	1	-	XferComplete event
OUT	Yes No	-	0	X	0	-	No event
OUT	Yes No	-	1	X	0	-	XferInProgress event
OUT	X Yes	X	X	X	X	0	XferComplete event <sup>1</sup>
OUT	X Yes	X	X	0	1	1	XferComplete event
OUT	X Yes	X	X	1	0	1	Search for CHN=0, accumulate ISP and IOC, then follow CHN=0 rules <sup>2</sup>
OUT	X Yes	X	1	0	0	1	XferInProgress event
OUT	X Yes	0	0	0	0	1	No event
OUT	X Yes	1	0	0	0	1	XferInProgress event

1. When a TRB receives whose CSP bit is 0 and CHN bit is 1 receives a short packet, the chained TRBs that follow it are not written back (for example, the BUFSIZ and HWO fields remain the same as the software-prepared value)
2. In the case of an OUT endpoint, if the CHN bit is set (and CSP is also set), and a short packet is received, the core retires the TRB in progress and skip past the TRB where CHN=0, accumulating the ISP and IOC bits from each TRB. If ISP or IOC is set in any TRB, the core generates an XferInProgress event. Hardware does not set the HWO bit to 0 in skipped TRBs. If the endpoint type is isochronous, the CHN=0 TRB will also be retired and its buffer size field updated with the total number of bytes remaining in the BD.

On XferComplete and XferInProgress events, status bits in the event indicate LST, IOC, Short Packet Received, or Bus Error status. In the "fast-forward" case, the IOC status is accumulated from the skipped TRBs, not just the TRB that received the short packet.

When the hardware writes back the TRBs, it updates the BUFSIZ field to represent the remaining unused buffer.

### 36.4.3.2.3.1 Number of TRBs Rule

- Software must set up only one TRB for a control setup or status stage.

- If software is preparing multiple transfers for an IN endpoint, it may be necessary to place a 0-length TRB between transfers that are MaxPacketSize aligned to indicate transfer boundaries to the host. This is not necessary, if the class driver and the host can handle bursting between transfers.
- If software is preparing multiple transfers for an OUT endpoint, it needs to place a MaxPacketSize TRB between transfers, if it expects the host to transmit a 0-length packet between transfers.

### 36.4.3.2.3.2 TRB Control Bit Rules

Transfer control bits must conform to the following restrictions:

For OUT endpoints, the CSP bit must be the same in every TRB within a Buffer Descriptor (either set or clear).

- The core autonomously checks the HWO field of a TRB to determine if the entire TRB is valid.
- Therefore, software must ensure that the rest of the TRB is valid before setting the HWO field to '1'. In most systems, this means that software must update the fourth DWORD of a TRB last. Every time software validates a TRB by setting HWO=1, it must also issue an Update Transfer command to the core.
- Software sets the HWO bit to 1 when it creates the TRB, and cannot modify it until hardware resets it to 0. However,
  - Software must detect when a "fast-forward" occurs on an OUT endpoint that receives a short packet, since some TRBs in a chain may still have their HWO bit set to 1 while belonging to software.
  - Hardware will not clear the HWO bit of a Link TRB. Therefore, software can only modify a Link TRB if the TRB prior to the Link TRB has its HWO bit set to 0.
- The LST bit must not be set to 1 for isochronous endpoints.
- For a Setup or Status TRB, set CHN=0, LST=1, and CSP=0.
- For the data stage of a control transfer, set CSP=0 in all TRBs and LST=1 in the last TRB of the data stage.
- For Link TRBs, the LST, CHN, IOC, ISP, CSP, and Stream ID fields are ignored and must be set to 0.
- The Link TRB's chain bit is implicitly equal to the chain bit of the TRB before it. If the TRB before it has CHN=1, then the Link acts as if it's CHN=1. If the TRB before it has CHN=0, then the Link acts as if it's CHN=0. A Link TRB cannot be the last TRB in a Buffer Descriptor.
- When CSP=1 and the core receives a short packet, it searches for the end of the Buffer Descriptor by finding the Normal TRB that has CHN=0. Therefore, it is illegal to setup a circular buffer with all Normal TRBs with CHN=1.

### 36.4.3.2.3.3 Buffer Size Rules and Zero-Length Packets

The hardware contains a cache that holds four TRBs per transfer.

For IN endpoints, the following rules apply:

- The number of chained TRBs necessary to construct a single packet must never exceed 3. A maximum of one Link TRB can be present in the chain.
- If software wants to indicate a transfer completion to the host by sending a zero-length packet after a multiple of MaxPacketSize, it must set up a zero-length TRB following the last TRB in the transfer.

For OUT endpoints, the following rules apply:

- If the first TRB has CHN=1, its buffer size must be  $\geq 1$  byte
- The total size of a Buffer Descriptor must be a multiple of MaxPacketSize
- A received zero-length packet still requires a MaxPacketSize buffer. Therefore, if the expected amount of data to be received is a multiple of MaxPacketSize, software should add MaxPacketSize bytes to the buffer to sink a possible zero-length packet at the end of the transfer.

For IN and OUT endpoints, the following rule applies:

- The BUFSIZ field in a Link TRB must be set to 0.

### 36.4.3.2.3.4 Transfer setup recommendations

Software can either set up transfers before the host attempts to move data on an endpoint ("preset" transfers) or can set up transfers on demand ("on-demand" transfers). When using preset transfers, software can safely disable the XferNotReady event in the endpoint configuration. However, when using on-demand transfers, the XferNotReady event must be enabled and software may not use the "No Response" variant of the Update Transfer command. The XferNotReady event is issued when the host attempts to move data on an endpoint when one of the following conditions is present:

- No previous transfer was started with Start Transfer.
- Not enough hardware-owned (HWO=1) TRBs are available to handle the requested data movement. The XferNotReady event must not be disabled for control endpoints because the event is an integral part of control transfer handling.

Although there are many valid ways to set up transfers, it is recommended that you choose one of three general mechanisms:

- When software wants to set up one transfer at a time and has the entire buffer available for transfer, it must set up TRBs that point to the data buffers and in the last

TRB it must set the LST bit and issue Start Transfer, which points to the first TRB location.

- When the USB transfer completes, the core will notify the software through the XferComplete event. The LST bit will be set in the status field of the event. The XferComplete will also release the Transfer Resource.
- A premature XferComplete event can happen before all the data buffers are exhausted, if there is a Bus Error or, in an OUT transfer a short packet has been received and CSP=0. During these conditions the TRB will have an updated BUFSIZ field which represents the amount of buffer remaining after the successful part of data transfer.
- Software can also set up an IOC bit TRB, so that it gets notified when data buffers are used up and can free them up sooner than waiting for the entire transfer to complete. On completing a TRB with IOC set, and not the last one, the core will issue an XferInProgress event with IOC set in the status field of the event.
- When software wants to set up one transfer at a time, but it has fewer data buffers available than the full transfer size, it must set up circular TRBs (using a Link TRB), and also set up IOC bits in the TRBs. Depending on buffer allocation and interrupt frequency it can set IOC for once in "x" number of TRBs.
  - As soon as the USB transfer for a TRB is completed, and if IOC is set, the core will generate an XferInProgress event with IOC set in the status field.
  - On seeing the event, software reuses the TRB and updates it with the next data buffer. It also issues an Update Transfer command, indicating it has updated a TRB.
  - If software is slow and hardware finds a TRB with HWO reset to 0, it waits for an Update Transfer command. Upon seeing the Update Transfer, it prefetches the TRB and continues the transfer.
  - When software reaches the end of the transfer, it sets the TRB LST bit. When hardware completes the TRB, it issues an XferComplete event and releases the Transfer Resource.
- When the Device software has multiple transfers to set up, it must set up circular TRBs and also set up CHN bits in all TRBs, except the last of each transfer. For OUT endpoints, multiple transfers can be supported when the CSP field is set to '0' if each transfer has a multiple of MaxPacketSize bytes, since a short packet will end the transfer. If multiple transfers may contain short packets, the CSP field must be set to '1' to enable the next transfer to continue even if a short packet is received.  
Depending on buffer allocation and interrupt frequency, it can either set the IOC for once in "x" number of TRBs or in the last TRB of each transfer.
  - For OUT endpoints, the transfer can finish prematurely due to a short packet from the host. In this case, the core processes the remaining TRBs, skipping the updates to these TRBs until it reaches TRB of the next transfer. While skipping

these TRBs, if any of the TRB it encounters the interrupt setting of ISP or IOC, it will generate the corresponding event once and ignore the interrupt settings of the remaining TRBs until a TRB of next transfer is reached.

- Device software can reclaim these skipped TRBs even though the HWO still indicates 1 (hardware-owned).
- For IN endpoints, software cannot stop providing transfers while it is ending transfers with CHN=0 and LST=0, otherwise it is possible that the endpoint is left in a flow-controlled state on the USB.

#### 36.4.3.2.4 Transfer Resource Usage and Transfer State

Software allocates one Transfer Resource for an endpoint during initialization. When software issues a Start Transfer command, this Transfer Resource is used. When an XferComplete event happens, the Transfer Resource is released back to software. Similarly, when End Transfer completes, the Transfer Resource is released.

Following is a typical transfer usage summary (not complete usage model):

- Software sets up data buffer(s) and TRB(s) in external memory with the TRB(s) pointing to the buffer(s).
- Software issues Start Transfer with the pointer to the first TRB in the command parameters.
- If software sets up all the buffers and TRBs at the same time (Host and Device Software negotiate the transfer size), then just one Start Transfer is enough, and software waits for a XferComplete event.
- If software did not set up all the TRBs, then every time software adds a new TRB (by setting HWO=1) it must issue an Update Transfer command. The hardware will fetch the TRB again and continue the transfer.
- The End Transfer command is used only during error conditions and not used during normal transfers. For example, if software has set up multiple transfers and if a USB Reset event happens, it must remove all the transfers from the queue using End Transfer with the ForceRM bit set 1.
- A transfer is completed when all the data has been transferred or a short packet has been received. If software is queuing in only single transfers at a time (normal method for mass storage - for ethernet over USB, software can set up multiple transfers using the Chain bit and Continue on Short Packet bit), once XferComplete interrupt has happened, the transfer is completed. When starting a new transfer, software now needs to issue the Start Transfer command.
- An OUT transfer's transfer size (Total TRB buffer allocation) must be a multiple of MaxPacketSize even if software is expecting a fixed non-multiple of MaxPacketSize transfer from the Host.

## Functional Description

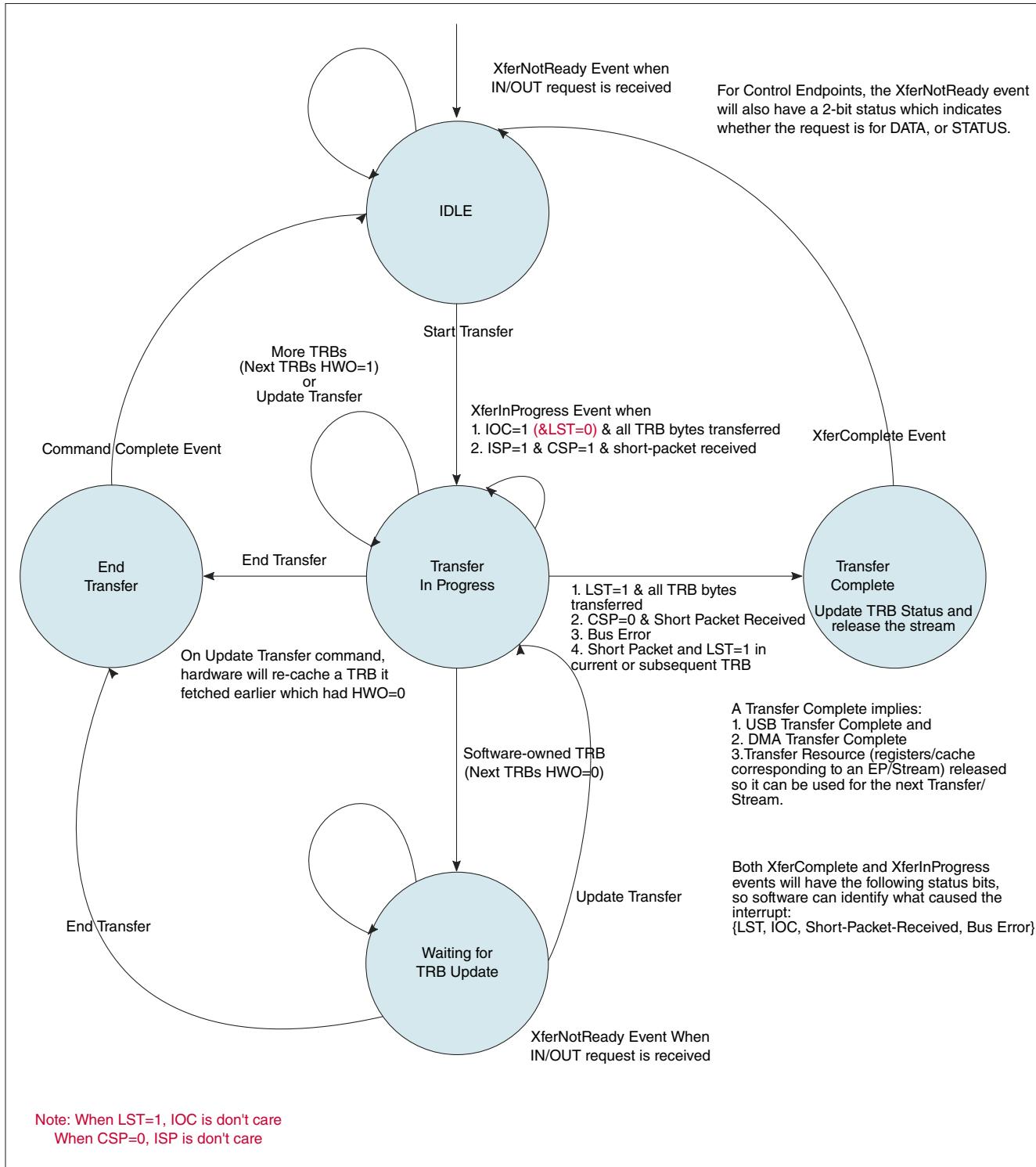


Figure 36-18. Transfer Usage State Machine.

### 36.4.3.2.5 Transfer Descriptions

The following sections describe typical transfer processes.

### 36.4.3.2.5.1 Non-Isochronous OUT Transfers

The sequence below describes a regular, non-isochronous OUT transfer (control-data/bulk/interrupt).

**Table 36-19. Non-Isochronous OUT Transfer Sequence**

Steps	Host	Device Core	Device Software
Software sets up data buffers before a host request			
Step 1	-	-	Software sets up Normal TRBs and enables DMA by issuing Start Transfer.
Step 2	-	Device fetches the TRB and caches it.	-
Step 3	Host sends data packets	-	-
Step 4	-	<p>Core receives the packets in RxFIFO and stores them in to TRB data buffers through DMAs. While the RxFIFO has space and the BUS DMA keeps up with the host data transfers, the device receives all the packets. Once all the data is received, the core updates the TRB status and then generates a XferComplete event.</p> <p>In SS mode, if GRXTHRCFG.USBRxPktCntSel is set to,</p> <ul style="list-style-type: none"> <li>• 0: the NumP sent depends on the DCFG.NumP and bMaxBurstSize</li> <li>• 1: the NumP sent depends on the RX FIFO space available when the packet is received</li> </ul> <p>When DMA is slow or CRC fails:</p> <ul style="list-style-type: none"> <li>• If the RxFIFO overruns or CRC error is detected, device times out in Non-SS mode or asks for the same data packet (through ACK/SeqNumber/Retry Bit) in SS mode.</li> <li>• In HS mode, on NAK the host will issue PING and device will ACK it (if there is enough space in the RxFIFO) and the host resumes with data again. In SS mode, the host will send the requested data. If the device receives any packet with other than the requested data sequence, it will ignore it.</li> </ul>	-
Step 5	-	-	Software processes the XferComplete event.
Software sets up data buffers after the host request.			
Step 6	Host sends data packets.	-	-
Step 7	-	Since DMA is not set up, the device will send NAK in Non-SS mode and will send NRDY in SS mode to the host. It also generates an XferNotReady event.	-
Step 8	-	-	On seeing the XferNotReady event, Software sets up TRBs and enables DMA by issuing Start Transfer.
Step 9	-	Steps 2-5 are followed.	-

*Table continues on the next page...*

**Table 36-19. Non-Isochronous OUT Transfer Sequence (continued)**

Steps	Host	Device Core	Device Software
Step 10	-	If the core encounters a short packet before the transfer size data programmed in the TRB is received, it skips the remaining TRBs of the transfer. If the core encounters control bits LST, IOC, or ISP it will generate events and, in the case of an LST TRB, the stream will be completed. Software has to reclaim any TRBs skipped with HWO=1.	On detecting a skipped TRB condition, software reclaims the TRBs with HWO=1.

### 36.4.3.2.5.2 Isochronous OUT Transfers

This section describes the difference between isochronous and non-isochronous OUT transfers referring to the sequence in [Non-Isochronous OUT Transfers](#)

- In Step 2, if the TRB is not fetched in time, the core discards the packet. In addition subsequent packets in the service interval also will be dropped.
- In Step 4, if DMA is enabled and if RxFIFO overrun happens or CRC fails, the core discards the packet. In addition subsequent packets in the service interval also will be dropped.
- In Step 7, if DMA is not enabled, the core drops the data and generates XferNotReady event.

In general, the TRB chain of a service interval will be released if the core receives DATA0 PID in HS USB 2.0 mode or receives a packet with LPF set in SuperSpeed mode.

Special scenarios are:

- If the core misses this last packet indication, it releases TRB at the service interval boundary.
- If the core encounters a short packet before the transfer size data programmed in TRB is received, it skips the remaining TRBs of the transfer.
- If the core encounters control bits 'LST' or 'IOC' or "ISP" it will generate event and, in the case of LST TRB, the stream will be completed, but software has to reclaim TRBs if any TRBs skipped with HWO=1.

### 36.4.3.2.5.3 Non-isochronous IN transfers

The sequence below describes a regular, non-isochronous IN data transfer (control-data/bulk/interrupt).

**Table 36-20. Non-isochrnoous IN transfers**

Steps	Host	Device Core	Device Software
-------	------	-------------	-----------------

*Table continues on the next page...*

**Table 36-20. Non-isochrrous IN transfers (continued)**

	Software sets up data buffers before a host request		
Step 1	-	-	Software sets up TRBs and enables DMA by issuing Start Transfer.
Step 2	-	Device fetches the TRB, caches it, and prefetches the data into TxFIFO.  Only in SS mode, the core sends ERDY (Async ERDY, even before host requesting data provided endpoint is in flow controlled state. LSP flow is same regardless of if the host asked for it.)	-
Step 3	Non-SS host issues a token request, SS host issues ACK TP to request data.	-	-
Step 4	-	If DMA fetches keep up with host requests, the device sends all the data, updates the TRB status, and then generates an XferComplete event when ACKs for all the packets are received.  When DMA is slow: <ul style="list-style-type: none"><li>• If the TxFIFO becomes empty on a host data request, the device sends a NAK in Non-SS mode. In SS mode, the device sends NRDY, and when data is available in TxFIFO it sends ERDY and waits for the host request again.</li><li>• If the TxFIFO becomes empty in the middle of a packet transfer (happens only in Threshold mode), in Non-SS mode the device will invert the CRC. In SS mode, it terminates with EDB after appending the corrupted CRC. The core waits for the host to retry the packet before fetching the data again.</li></ul> If the host retries a packet: <ul style="list-style-type: none"><li>• In Non-SS mode, a transmitted packet is kept in the TxFIFO until the ACK arrives (this is immediate in Non-SS mode). If the host retries a packet, it is immediately sent out from the TxFIFO.</li><li>• In SS mode, before ACK can be received, the core could have sent additional packets in a burst. When a retry happens, NRDY is sent to the host and the TxFIFO is flushed. A DMA command is issued on the master bus to prefetch data from where the host is asking. Once data is fetched into the TxFIFO, the device sends ERDY and waits for host request again.</li><li>• In Threshold mode, when the TX threshold amount of data is fetched, the device sends ERDY to the host.</li></ul>	-
Step 5	-	-	Software processes XferComplete event.
	Software sets up data buffers after a host request.		

*Table continues on the next page...*

**Table 36-20. Non-isochrrous IN transfers (continued)**

Step 6	Non-SS host issues a token request. SS host issues ACK TP to request data.	-	-
Step 7	-	Since DMA is not set up, the device sends a NAK in Non- SS mode, or NRDY in SS mode, to the host. The core also generates an XferNotReady event	-
Step 8	-	-	On seeing the XferNotReady event, software sets up Normal TRBs and enables DMA by issuing Start Transfer.
Step 9	-	Steps 2-5 are followed.	-
Step 10	-	When software knows it is talking to a host that always ends transfers with short packets, and therefore always requires a zero-byte packet to end a transfer, it includes a zero-byte TRB (CHN=0) after the last TRB which is an even multiple of MaxPacketSize.  When software does not know if the host is going to do another IN token after receiving the exact number of bytes it expected (and it was a multiple of MPS), it does not include a zero-byte TRB, and sets LST in the last TRB. If the host returns with another IN for a zero-byte packet, it uses the XferNotReady/Start Transfer mechanism to add one zero-byte TRB. This is called "On- Demand."	-

#### 36.4.3.2.5.4 Isochronous IN Transfers

This section describes the difference between isochronous and non-isochronous IN transfers referring to the sequence in [Non-isochronous IN transfers](#).

- In Step 4, if the Tx FIFO is empty when host IN request arrives, the core returns a zero length packet in both Non-SS and SS modes.
- In Step 7, the device returns zero length packet and generates an XferNotReady event.

Special scenarios:

- When data is fetched and there is no request from the host (possibly due to a missing request or corrupted request) the data will be flushed at the end of the service interval (corresponding Micro- Frame) boundary and the core moves onto fetching next service interval data. If there are some TRBs that are not serviced in the service interval, the core will skip these TRBs.
- If the core encounters control bits LST or IOC, it will generate an event, and in the case of LST TRB, the stream will be completed, but has to reclaim TRBs if any TRBs skipped with HWO=1.

### 36.4.3.2.6 Handling ENDPOINT\_HALTI

On receiving a SetFeature (ENDPOINT\_HALTI) control transfer, software issues a Set Stall command on the endpoint. Because of the delay from when software issues the command and when it is recognized by the core, other transfer events (XferInProgress, XferComplete) may be received prior to the endpoint being halted. For a description of the Set Stall endpoint command and its effects, see [Commands 4 and 5: Set Stall and Clear Stall \(DEPSSTALL, DEPCSTALL\)](#).

On ClearFeature (ENDPOINT\_HALTI), software must first remove all pending transfers for the endpoint through the End Transfer command. It may then issue a Clear Stall command on the endpoint followed by Start Transfer to start transfers again. For a description of the Clear Stall endpoint command and its effects, see [Commands 4 and 5: Set Stall and Clear Stall \(DEPSSTALL, DEPCSTALL\)](#).

### 36.4.3.2.7 Handling L1 Event During a Transfer

This section discusses the scenario in which the software already issued the Start Transfer command but the LPM occurred before or during the data transfer.

When an IN transfer is in progress, (that is, started and not completed yet), and the software sees a link state change event to L1, then the software can use the GDBGIFOSPACE register (write followed by a read) to determine if the TxFIFO is empty or not. If the TxFIFO is not empty, it can initiate a remote wakeup.

#### NOTE

- The GDBGIFOSPACE register is a debug register which gives the 'Space Available' for the endpoint selected with a write prior to the read. If this space available is less than the particular TxFIFO's maximum size, then the TxFIFO is not empty.
- Software can enable the Link State change event for USB 2.0 as there are not a lot of events compared to SS.

### 36.4.3.3 Isochronous Transfer Programming Model

Isochronous endpoints get guaranteed service by the host during a "Service Interval". The service interval for an endpoint is communicated to the host via the endpoint descriptor, and it is configured in the hardware through the DEPCFG command. However, unexpected behavior on the bus may prevent the host from servicing the endpoint as frequently as expected.

Although the endpoint receives "guaranteed" service during the interval, there are no bus-level acknowledgments, which means that there is no guarantee that the host (IN endpoints) or the device (OUT endpoints) receives correct data.

Therefore, the isochronous programming model differs from the bulk programming model to accommodate these two factors:

- No bus-level acknowledgement of packet transmission or reception
- No guarantee that all the data setup for an interval has been transmitted or received

For isochronous endpoints, software has the following responsibilities:

- Set up enough TRBs to keep up with the rate of data transmission or reception
- For FIFO-based IN endpoints, guarantee the validity of the TX data at least 1 micro Frame before the beginning of the interval that the data will be transmitted

Hardware has the following responsibilities:

- Transmit or receive data at the appropriate bus time
- For FIFO-based IN endpoints, delay fetching until 1 micro Frame before the beginning of the interval that the data will be transmitted
- Maintain a 1-to-1 correspondence between Buffer Descriptors and the intervals on the bus
- Report missed isoc intervals and update the buffer sizes in TRBs to reflect the amount of data moved

#### **36.4.3.3.1 Definitions**

- bInterval: The field in an endpoint descriptor used to communicate the service interval of the device. Its value ranges from 1 to 16, however, the USB 3.0 core supports a range of 1 to 14.
- Service interval: An integral number of microframes ( $2^{[b\text{Interval}-1]}$ ) during which the host will poll the device to move data.
- Beginning of interval: The interval begins when the least significant (bInterval-1) bits of the bus time are all 0's.
- End of interval: The interval ends when the least significant (bInterval-1) bits of the bus time are all 1's.
- Data payload: The number of bytes to be moved during the service interval. It will be transmitted using MaxPacketSize packets.
- Buffer descriptor: A set of one or more TRBs with CHN=1 and the last one having CHN=0. The group of TRBs represents the data buffers for one service interval.
- Last TRB of a Buffer Descriptor: The TRB within a Buffer Descriptor that has CHN=0. When an interval has completed, this TRB will contain the total remaining buffer size of the Buffer Descriptor and also the completion status of the interval.
- Microframe ( $\mu\text{F}$ ): A unit of time on the USB that lasts 125  $\mu\text{s}$ .

- Start of Frame (SOF): The beginning of a microframe.
- FIFO-Based Isoc IN: A sub-type of Isoc IN endpoints that has the characteristic of always assuming the HWO bit in a TRB is '1' and delays fetching of transmit data.
- Prefetch delta: For FIFO-based IN endpoints, TX data may be prefetched as early as  $1 \mu\text{F}$  before the beginning of a Service Interval, but never earlier.
- Retire a TRB: When HW sets the HWO bit to 0 in a TRB and writes it back.
- Retire a Buffer Descriptor: When HW retires at least the first and last TRB of a Buffer Descriptor.
  - Other TRBs of the Buffer Descriptor may not be retired if an interval was missed (IN or OUT) or if a short packet/last packet is received (OUT). In this case, software may reclaim those TRBs even though HWO=1.
- Missed interval: When the device does not move all the data in an interval. This may occur when the host does not poll for all the data, or because of application-side delays that prevent all the data from being moved.
  - For IN endpoints, this occurs when the host does not request all the data prepared in the Buffer Descriptor, or not all the data is fetched from the system bus in time.
  - For OUT endpoints, this occurs when the host does not send a packet, a packet is dropped due to CRC error, or the system bus is too busy to accept the data.

### 36.4.3.3.2 Endpoint configuration

An isochronous endpoint is setup in much the same way as a bulk endpoint, with the following exceptions:

- The bInterval\_m1 value in DEPCFG Parameter 1 must be set to the value reported in the endpoint descriptor minus 1. When the core is operating in Full Speed, bInterval\_m1 must be set to 0.
- The MaxPacketSize value in DEPCFG Parameter 0 must be set to the same value reported in the endpoint descriptor.
- The "FIFO-based" bit in DEPCFG Parameter 1, bit 31, must be set for FIFO-based isochronous endpoints.
- For the best performance, set the TXFIFOs corresponding to Isochronous IN endpoints to a high priority.

After the endpoint is configured, it can be enabled by setting the appropriate bit in the DALEPENA register.

### 36.4.3.3.3 Transfer configuration

Software describes isochronous transfers through a series of Buffer Descriptors. Each Buffer Descriptor corresponds directly to one service interval of data. The fields within an isochronous TRB are the same as bulk TRBs with the following exceptions:

- The SOF field in an IN endpoint TRB is a don't care. For an OUT endpoint, the core will write this field with the timestamp of the last packet received into the TRB's buffer. This information is not needed for normal operation, only for debug purposes.
- For High-Speed, High-Bandwidth IN endpoints, a maximum of 3 packets can be sent during an interval. The PktCntM1 field ([25:24] of the 3rd DWORD) must be set to the (number of packets in the Buffer Descriptor - 1). For example, if 3 packets are to be transmitted during the interval, this field must be set to 2. For Super-Speed and OUT endpoints, this field is a don't care.
- The first TRB in a Buffer Descriptor must have the TRBCTL field set to the "Isochronous-First" type while all others have this field set to "Isochronous".
- The ISP bit is renamed IMI (Interrupt on Missed Interval) and should be set if software wants to receive an XferInProgress event when at least 1 packet is missed within an interval.
- The IMI bit should be set to the same value in all TRBs of a Buffer Descriptor.
- The CSP bit must be set to 1 (short packets cause the hardware to move to the next Buffer Descriptor, they do not end an isochronous transfer).
- The LST bit should be set to 0 (isochronous transfers normally continue until the endpoint is removed entirely, at which time an End Transfer command is used to stop the transfer).

All other bits and fields (IOC, HWO, CHN, BPTR, BUFSIZ) retain the same behavior as they have for bulk endpoints. If software needs to receive an interrupt after every service interval, it should set the IOC bit to '1' in each TRB. However, if software wants to reduce the interrupt frequency and the application can tolerate some latency, the IOC bit can be set to '0' and the core will not generate a XferInProgress event when the TRB is completed.

For OUT endpoints, the Buffer Descriptor must describe a total buffer size of:

$$(\text{Mult}+1) * \text{MaxBurst} * \text{MaxPacketSize}$$

### 36.4.3.3.4 Starting a Transfer

The hardware will report the bus time that the host starts polling the endpoint inside the XferNotReady event. Software will use this value as a reference when issuing the Start Transfer command to serve as time synchronization with the host.

1. Set up TRBs and data buffer for the endpoint.
2. After configuring and enabling the endpoint, wait for an XferNotReady event.

The XferNotReady event will contain the time that the host started polling the endpoint in the upper 16 bits.

3. Issue the Start Transfer command to the endpoint with a future microframe time written into the upper 16 bits of the DEPCMD register. The future microframe time must be a value that is an integral multiple of intervals after the time reported in the XferNotReady event and aligned to the beginning of an interval. For example, if bInterval is 3 (4 microframes), and the XferNotReady time is 2, the value can be 4, 8, 12, and so on. The future microframe time must also be no greater than 4 seconds past the time reported in the XferNotReady event.
4. If the future microframe time has already passed when the command is received, the core will respond with an error (bit 13 in the Command Complete event).

In this case, software must issue End Transfer, then wait for another XferNotReady event and attempt the command again with a time that is further in the future. Otherwise, the first Buffer Descriptor will be used for the interval starting with the microframe time specified in the Start Transfer command.

#### **36.4.3.3.5 Core Behavior During an Interval**

After a transfer has been started, the hardware will perform the following functions for IN endpoints:

1. Fetch TX data as early as one interval prior to the beginning of the interval (call it A) if the HWO bit is set to one in the TRB.
2. Decrement the buffer size of each TRB as packets are transmitted.
3. Retire TRBs when their buffer size has reached 0, issuing an XferInProgress event if the IOC bit is set.
4. If the next interval (B) starts before all the packets have been transmitted for interval A:
  - a. Flush the TxFIFO.
  - b. Retire the Buffer Descriptor of interval A with a "Missed Isochronous" status.
  - c. Retire the Buffer Descriptor of interval B with a "Missed Isochronous" status.
  - d. See [Checking interval status](#) for a description of how software can determine that an interval ended unexpectedly.
  - e. Go to step 1 to prepare for interval C
5. Otherwise, if all the TRBs of interval A are completed, the hardware will prepare for interval B.
6. If the host completes an interval by polling for all the data, but then it polls the endpoint again during the same interval, the hardware will respond with a zero-length packet and no interrupt will be made to software.

7. If the host polls the endpoint prior to the time specified in the Start Transfer command, the hardware will respond with a zero-length packet and no interrupt will be made to software.
8. If the host polls the endpoint during the expected interval and the hardware has no data prepared, the core will respond with a zero-length packet, but no XferNotReady event will be generated because the transfer is active. When the next interval starts, the XferInProgress event is generated (based on ISP and IMI) with the "Missed Isochronous" status, which is the way software finds out that this occurred.

For OUT endpoints, the hardware performs the following functions:

1. If a packet is received for the correct interval represented by the Buffer Descriptor (call it A).
  - a. Write the packet into the buffer.
  - b. Decrement the buffer size of the TRB.
  - c. Write the timestamp of the received packet into the TRB.
  - d. Retire a TRB when its buffer size reaches 0, issuing an XferInProgress event if the IOC bit is set.
2. If a short packet or packet with the last packet flag (lpf) is received for the correct interval.
  - a. Write the packet into the buffer.
  - b. Retire the Buffer Descriptor of interval A with the TRBSTS set to 0.
  - c. If the IOC bit is set in the last TRB of the Buffer Descriptor, issue an XferInProgress event with bit [13] set.
  - d. Go to step 1 to prepare for interval B.
3. If a packet is received for the correct interval but it has an error (CRC error, RxFIFO overflow), the core will not increment its expected sequence number value which causes future packets within the same interval to be dropped.
4. If a packet is received at a bus time prior to the time specified in the Start Transfer command, it will be dropped.
5. If a packet is received for the correct interval (A), but not enough buffer space is available for the core to write the packet (due to the HWO bit still '0' in one or more of the TRBs of the Buffer Descriptor), no XferNotReady event will be generated. The core will behave as follows:
  - a. Wait until software sets the HWO bit to '1' and issues an Update Transfer command
  - b. Retire the Buffer Descriptor of interval A with a "Missed Isochronous" status
  - c. Go to step 1 to prepare for interval B.
6. If a packet is received for the next interval (B) before all the packets have been received for interval A:
  - a. Retire the Buffer Descriptor of interval A with a "Missed Isochronous" status.
  - b. Retire the Buffer Descriptor of interval B with a "Missed Isochronous" status.

- c. See section [Checking interval status](#) for a description of how software can determine that an interval ended unexpectedly.
- d. Go to step 1 to prepare for interval C.

### Special Considerations for Isochronous OUT Endpoints

For OUT isochronous endpoints, the hardware detects a missed interval when the host sends a data packet in a future interval. It does not detect the missed interval at the exact interval boundary. Therefore, if the host abruptly stops sending isochronous OUT packets, there will be no interrupt or event indicating this to software. Software should use one of the following mechanisms to detect the interruption of isochronous OUT traffic:

1. The ultimate consumer of the isochronous OUT data will detect an underflow.
2. The device driver can use a timer detect that no XferInProgress events have been received for multiple intervals.

If this occurs, software should issue an End Transfer command to the endpoint and wait for a XferNotReady event which signals that the host is ready to resume isochronous traffic.

#### 36.4.3.3.6 Checking interval status

As packets are transmitted or received during an interval, the buffer size of each TRB will be decremented. When the host is operating normally and polling for all the interval data, the buffer size of all the TRBs of an IN endpoint Buffer Descriptor will be zero after the interval has completed. For OUT endpoints, if the host sends less data than the Buffer Descriptor was setup for, the remaining buffer size may be greater than 0.

When the interval completes, the final remaining buffer size and missed isoc status is written to the last TRB of the Buffer Descriptor. If MissedIsoc is set, then it means the BUFSIZ is not accurate and may indicate that more data was transmitted or received than in reality. If the MissedIsoc is not set, it means the BUFSIZ field is correct.

When hardware detects the end of an interval (including normal and abnormal ends), it performs the following:

- If it has not already been retired, the first TRB of the Buffer Descriptor will be retired.
- Any non-first TRBs with CHN=1 that had not already been retired will not be written back. HWO will still be 1, and software can reclaim them for another transfer.
- The last TRB of the Buffer Descriptor will be retired with:
  - HWO = 0.

- BUFSIZ = The total remaining buffer size of the Buffer Descriptor.
- TRBSTS = "Missed Isoc" if any packets were missed, zero otherwise.
- If the IOC bit is set in the last TRB, or the IMI bit is set and packets were missed, hardware will issue an XferInProgress event

Software can get notification of an interval completing by setting the IOC bit in the last TRB of the Buffer Descriptor. The IOC bit may also be set in any other TRB of the Buffer Descriptor if software wants earlier notification that a TRB has completed.

Software can also get only a notification of an interval completing unexpectedly by setting the IMI (Interrupt on Missed Isoc) bit in the last TRB of the Buffer Descriptor. When an interval completes unexpectedly and either the IOC or IMI bit is set in the TRB, the MissedIsoc bit (15) of the XferInProgress event will be set. The following table shows which events are generated in each scenario:

**Table 36-21. TRB event generation events**

Scenario	IOC	IMI	Action
TRB completed with a MaxPacketSize packet	0	0	No interrupt
	0	1	No interrupt
	1	0	XferInProgress (IOC=1)
	1	1	XrefInProgress (IOC=1)
TRB completed with a short packet	0	0	No interrupt
	0	1	No interrupt
	1	0	XferInProgress (Short=1, IOC=1)
	1	1	XrefInProgress (Short=1, IOC=1)
Interval completed due to missed packet (missed isoc)	0	0	No interrupt
	0	1	XferInProgress (MissedIsoc=1, IOC=1)
	1	0	XferInProgress (MissedIsoc=1, IOC=1)
	1	1	XrefInProgress (MissedIsoc=1, IOC=1)

It is normal to lose two intervals at a time when an error occurs during one interval. One interval is lost because of the host (which may not be polling enough) and the second interval is dropped because the device needs time to synchronize up to the next (third) interval.

### 36.4.3.3.7 Adding Intervals to a Transfer

Because isochronous endpoints represent a stream of data, the TRBs of an isochronous endpoint will normally be setup in a circular list, such as:

- TRB 1 (CHN=0, IOC=1)
- TRB 2 (CHN=0, IOC=1)
- Link (to TRB 1)

Example: Assume TRB 1 represents the data for the first interval and TRB 2 represents the data for the second interval. Because IOC=1, when the core completes the first interval, it will issue an XferInProgress event which indicates to software that TRB 1 can be analyzed to retrieve the results from interval 1. TRB 1 can be re-used to represent the data for the third interval by setting HWO=1 and issuing an Update Transfer command.

Each time software sets up TRBs for a new interval, it must follow the guidelines in the Transfer Configuration section. Software will set up TRBs for future intervals when it obtains those buffers from another software layer or when TRBs are retired by the core.

### 36.4.3.3.8 Moderating Events

During a transfer, software may enable or disable any endpoint-specific event by re-issuing the DEPCFG command with the "Config Action" set to "Modify" and a modified DEPEVTEN field. All other field. All other fields in all other parameters must remain the same as the initial endpoint configuration.

### 36.4.3.3.9 Other Types of Isochronous Endpoints

The above description of the isochronous endpoint programming model assumes that software is setting up buffers in external memory that apply to certain intervals of data. This is called a "buffer-based" isochronous endpoint.

However, there is another mode that the hardware supports to accommodate other mechanisms of sourcing or sinking isochronous data, called "FIFO-based" and the types are defined by bits [31:30] of DEPCFG Parameter 1. This table illustrates the differences between the models:

**Table 36-22. Isochronous Endpoints Models**

Type	FIFO-Based [31]	Bulk-based [30]	TRB Fetch	TRB Write back	TX DMA	Retire Buffer Descriptor at passed interval
Buffer-based	0	0	When HWO=1	Yes	When HWO=1, no earlier than one interval ahead of time.	Yes
FIFO-based	1	0	When internal cache space available	No	No earlier than 1 $\mu$ F before the interval	Yes
Invalid	1	1				

### 36.4.3.3.9.1 FIFO-based isochronous IN Endpoints

The software of some applications is unable to keep up with the short latency and high bandwidth of isochronous traffic and require the data to be sourced and done a sync through external FIFOs. One example is that there are two external TxFIFOs, one for each interval, and an address-to-FIFO-pop logic is implemented so that when the core attempts to read from interval 1's address, the translation logic converts this into a pop signal for interval 1's TxFIFO. When the core attempts to read from interval 2's address, the translation logic converts this into a pop signal for interval 2's TxFIFO.

To describe a FIFO-based implementation, software sets up an endpoint with the "FIFO-based" configuration bit set. This type of endpoint ignores the HWO bit in all TRBs, assuming that the TRB is always valid, and that the core should never write it back. Software chooses the buffer pointers within the TRBs to correspond to the address needed by the translation logic to specify which FIFO should be popped. By also using the CHN bit, headers and payload can be concatenated from different FIFOs. For example, if there are 4 external FIFOs: HA, PA, HB, PB, where HA/HB contain the 4 byte header for 2 intervals and PA/PB contain the 512 byte payload for 2 intervals, this can be described by using the following 5 TRBs:

1. BUFPTR=HA, IOC=0, CHN=1, BUFSIZ=4
2. BUFPTR=PA, IOC=1, CHN=0, BUFSIZ=512
3. BUFPTR=HB, IOC=0, CHN=1, BUFSIZ=4
4. BUFPTR=PB, IOC=1, CHN=0, BUFSIZ=512
5. Link to (1)

Every interval, the core will be creating a 516 byte packet that consists of 4 bytes from the header FIFO and 512 bytes from the payload FIFO. However, if the host does not poll for the packet, the core will skip 1 or 2 intervals of popping, depending on whether it has already started reading the next interval. External logic (or software) is responsible for flushing and refilling alternate FIFOs so that the core is always reading the correct data for the next interval. In normal intervals, the FIFOs will be empty, but when an interval is missed, there may still be data present in the FIFOs.

#### Software Requirements:

- The "FIFO-based" bit must be set in the DEPCFG when configuring the endpoint.
- No field within any TRB may be changed after the Start Transfer command is issued.
- Data must be valid in the external FIFO at least 1  $\mu$ F before the beginning of the interval for which the data is intended.

#### Core Behavior:

- The earliest the core will read from the FIFO will be 1  $\mu$ F before the beginning of the interval for which the data is intended.

- The core will not write back the TRB after it has completed it.
- The core will re-read the TRB from external memory even though it is not allowed to change.
- The only indication of missed intervals is the XferInProgress event if IOC or IMI is set in the TRBs. No indication will be made of how much data was actually transmitted.

#### **36.4.3.3.9.2 FIFO-based isochronous OUT Endpoints**

The FIFO-based model is also supported for OUT endpoints. In this case, core writes to specific addresses will be translated into external FIFO pushes. The same example from above (4 byte headers and 512 byte payloads) can be applied to OUT endpoints where the headers are split from the payload as the core receives 516 byte packets. The same software requirements and core behavior apply (for example, no writebacks and no indication of how much data was received). External logic assumes that if less than the expected amount of data is pushed into an interval's FIFO, the interval is invalid.

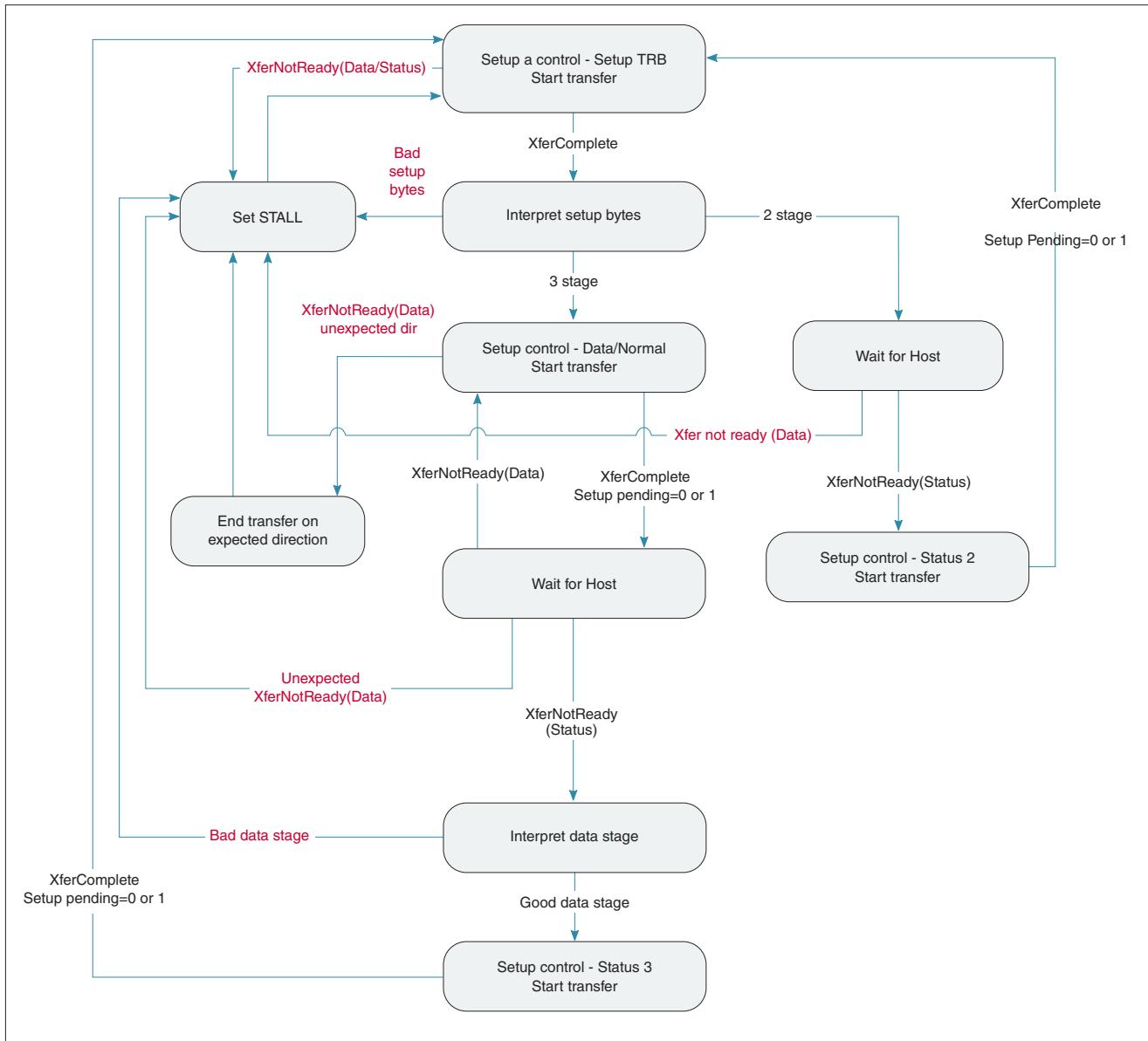
#### **36.4.3.3.10 End a Transfer**

Since isochronous endpoints have no bus-level acknowledgements, it is not necessary to set LST=1 in a TRB to gracefully end an isochronous transfer. Software can issue an End Transfer command to end an isochronous transfer. The core will wait until it can complete operations for the endpoint before returning the Command Complete event in response.

#### **36.4.3.4 Control transfer programming model**

Control transfers follow a flow of setup/data/status. On the USB bus, there are many error scenarios that can disrupt this flow. The hardware has a special mechanism that automatically recovers from these scenarios as long as software follows this single control transfer programming model.

## Functional Description



**Figure 36-19. Software flow for control transfers**

If hardware does not receive a setup packet, it discards the unexpected data and status stages. It does not issue XferNotReady to the application. The hardware will also not generate a XferNotReady event if it receives a setup packet before a TRB is prepared for it. As control endpoints are bi-directional, the IN and OUT directions of the endpoint are used for the different stages as follows:

### Control read:

- EP0 - Setup OUT
- EP1 - Data IN
- EP0 - Status OUT/Status TP

## Control write or 2-stage transfer:

- EP0 - Setup OUT
- EP0 - Data OUT (if present)
- EP1 - Status IN/Status TP

Set STALL is always issued on EP0, and the XferComplete/XferNotReady events are generated on the correct direction. For other control endpoints, substitute EP0 with the OUT direction (such as, 2, 4, 6) and EP1 with the IN direction (such as, 3, 5, 7). The same programming model is used for USB 2.0 and USB 3.0 control transfers.

### 36.4.3.4.1 Two-stage control transfer programming model

1. Software sets up a Setup TRB and issues Start Transfer on EP0 pointing to the Setup TRB.
2. After XferComplete is received, software interprets the setup bytes. If they are bad, issue Set Stall on EP0 and go back to the step 1. If a XferNotReady (Data/Status) event is received before the XferComplete event for the Setup stage, issue Set Stall. This is an error case where the host is attempting to move data or start the status stage for a previous control transfer that has already completed.
3. Wait for an XferNotReady event for the Status stage which will occur on EP1. If an XferNotReady event for the Data stage is received (either direction), issue Set Stall on EP0 and go back to the step 1. This is an error case where the host is attempting to start the data stage when the setup bytes did not indicate a data stage was present.
4. After XferNotReady (Status) is received, start the status stage by issuing Start Transfer on EP1 pointing to a valid Status-2 TRB.
5. After XferComplete is received on EP1, go back to the step 1.

Although not required, software may look at the Status TRB and examine the Setup Pending bit in the TRB status field. If it is set, it means that this control transfer was aborted on the USB bus and that the host did not receive the Status stage ACK.

### 36.4.3.4.2 Three-stage control transfer programming model

1. Software sets up a Setup TRB and issues Start Transfer on EP0 pointing to the Setup TRB.
2. After XferComplete is received, software interprets the setup bytes. If they are bad, issue Set Stall on EP0 and go back to the step 1. If a XferNotReady (Data/Status) event is received before the XferComplete event for the Setup stage, issue Set Stall. This is an error case where the host is attempting to move data or start the status stage for a previous control transfer that has already completed.
3. Start the data stage by issuing Start Transfer on EP0 (control write) or EP1 (control read) pointing to a valid Control-Data TRB.

- a. If an XferNotReady (Data) event is received for the incorrect direction, software must issue an End Transfer for the data stage it has already started, then issue Set Stall. This is an error case where the host is attempting to move data in the wrong direction.
- b. If a XferNotReady (Data) event is received for the correct direction, ignore the event and continue to wait for a XferComplete event.
4. After the XferComplete event is received, software waits for an XferNotReady event. Although not required, software may look at the completed Data TRB(s) and examine the Setup Pending bit in the TRB status field and the BUFSIZ. If the Setup Pending bit is set or the BUFSIZ is non-zero, it means that this control transfer was aborted on the USB bus and that the host did not complete the data stage.
5. If a XferNotReady (Data) event is received after the XferComplete for the Data stage, it could mean one of two things:
  - a. This host is trying to complete the data stage by moving a 0-length packet. This can occur if the data stage was an exact multiple of max packet size. If this is the case, software sets up an extra TRB (with a BUFSIZ of max packet size for control writes, or 0 for control reads), issues Start Transfer, and goes back to the step 4.
  - b. This host is trying to move more data than specified in the wLength field of the setup bytes. In this case, software issues Set Stall on EP0 and goes back to the step 1.
6. When the XferNotReady (Status) event is received, software interprets the data stage (for example, number of bytes transferred, content of the control write buffer) and decides if the data stage was successful. If not, it issues Set Stall on EP0 and goes back to the step 1.
7. Start the status stage by issuing Start Transfer on EP1 (control write) or EP0 (control read), pointing to a valid Status-3 TRB.
8. After XferComplete is received, go back to the step 1.

Although not required, software may look at the Status TRB and examine the Setup Pending bit in the TRB status field. If it is set, it means that this control transfer was aborted on the USB bus and that the host did not actually receive the Status stage ACK. When device software sets STALL with the Set Stall command, it is possible that this command can be delayed on the chip bus due to latency or the chip bus servicing other agents. In this case, software can receive an XferNotReady event if the core detects no TRB active before it detected the Set Stall. Software must service these events knowing that it stalled the control command.

### 36.4.3.4.3 Handling fewer requests than wLength

In a control IN transfer, if the host asks for less data than wLength, the core skips the TRBs of the data stage and generates an XferComplete event after encountering the TRB that has LST=1. Hardware sets HWO to 0 in those TRBs that it skipped and in the last TRB. In a control OUT transfer, if the host sends less data than wLength, the core treats this like a short packet received into a TRB with CSP=0. It immediately generates a XferComplete event, not setting the HWO bit in the remaining TRBs to 0. Software needs to reclaim those TRBs that still have HWO=1.

### 36.4.3.4.4 Control OUT transfer examples

The sequence in the following table describes two and three stage control OUT transfers.

**Table 36-23. Control OUT transfer**

Steps	Host	Device core	Device software
<b>Setup stage</b>			
1	-	-	Software sets up the Setup TRB and enables DMA by issuing Start Transfer to EP0 (endpoint 0, OUT direction).
2	-	Device fetches the TRB and caches it.	-
3	Host sends SETUP packet.	-	-
4	-	SETUP packet is received in the Rx FIFO. There are 24 bytes allocated for each control endpoint, which guarantees reception of the three back-to-back SETUP packets. An ACK handshake is returned in non-SS mode. An ACK TP with NumP=1 is returned in SS mode. After SETUP, bytes are transferred through the DMA, and the core issues an XferComplete event.	-
5	-	If the host starts a Data/Status stage, the core returns NAK in Non-SS mode or NRDY TP in SS mode. If the SETUP request is invalid, core returns STALL (when software sets STALL) in non-SS mode. In SS mode the core returns ERDY, and when the Data/Status stage is received it returns STALL.	Software processes the XferComplete event and decodes the SETUP bytes. If a Data stage is required, it will set up one Control-Data TRB and possibly subsequent Normal TRBs, then issue Start Transfer to EP0 (endpoint 0, OUT direction). If a Data stage is not needed, it will skip to the Status stage.  If the SETUP request is invalid, software instead sets the STALL bit by issuing Set Stall command for EP0 (OUT direction).
<b>Data stage</b>			
6	-	Device fetches the TRB and caches it. In SS mode only, the core will send ERDY.	-

*Table continues on the next page...*

**Table 36-23. Control OUT transfer (continued)**

Steps	Host	Device core	Device software
7	Host sends data packets.	-	-
8	-	The NumP for control transfer will be 1.  In addition, if another SETUP is received before the data transfer happens on USB, the core will skip the data transfer that software sets up without waiting for it to happen on the USB bus, and send XferComplete (when LST=1 TRB encountered) with the SetupPndg status bit set. Software has to reclaim the TRBs with HWO=1 in the skipped TRBs.	-
9	-	If the host starts a Status stage, the core returns NAK in Non-SS mode or NRDY TP in SS mode.	On seeing the event: <ul style="list-style-type: none"> <li>Software decodes the control write data. If the Data stage is OK, then it will wait for XferNotReady with "Control Status Request" set, set up a Control-Status-3 TRB (0-bytes), and enable DMA by issuing Start Transfer to EP1 (endpoint 0, IN direction).</li> <li>If the Data stage is not OK, then software sets STALL by issuing Set Stall command to EP0.</li> </ul>
<b>Status stage</b>			
10	-	Device fetches the TRB and caches it. In SS mode only, the core will send ERDY	Software sets up a Status stage TRB only after the Data stage of a three-stage control transfer is completed and an XferNotReady event occurred.
11	Host requests status packets.	-	-
12	-	In Non-SS mode, the device will send a zero-length packet if STALL is not set; otherwise, it will send STALL.  In SS mode, if STALL is not set, then it sends ACK TP with NUMP=1 else it sends STALL.  If another SETUP is received before the status transfer on USB, the core will send XferComplete with the SetupPndg status bit set.  The STALL bit will be cleared by the core whenever it receives a SETUP packet. SETUPS are always accepted.	-
13	-	-	On seeing the event: <ul style="list-style-type: none"> <li>If there are no other errors, then the transfer has completed successfully.</li> </ul>

### 36.4.3.4.5 Control IN transfer examples

The sequence in the following table describes two and three stage control IN transfers.

**Table 36-24. Control IN transfer**

Steps	Host	Device Core	Device Software
<b>Setup stage</b>			
1-5	Steps 1-5 are same as <a href="#">Control OUT transfer examples</a> , except that software sets up Tx buffers and Start Transfer is issued to EP1 (endpoint 0, IN direction).		
<b>Data stage</b>			
6	-	Device fetches the TRB and caches it. It prefetches the data and puts it in the TxFIFO. The core sends ERDY only in SS mode.	-
7	Host requests data packets	-	-
8	-	This step is similar to the step 4 of <a href="#">Non-isochronous IN transfers</a> . The NumP for control transfer is 1. In addition, if another SETUP is received before the data transfer software has set up without waiting for it to happen on the USB bus and send XferComplete (when LST=1 TRB encountered) with the SetupPndg status bit set. Software has to reclaim the TRBs with HWO=1 in the skipped TRBs and flush the TxFIFO.	-
9	-	If the host starts the Status stage, the core returns NAK in non-SS mode or NRDY TP in SS mode.	On seeing the event: <ul style="list-style-type: none"> <li>• If the Data stage is OK, then it sets up Control-Status-3 TRB (zero-bytes) and enables DMA by issuing Start Transfer to EP0 (endpoint 0, OUT direction).</li> <li>• If the Data stage is not OK, then software will set a STALL by issuing Set Stall command to EP0 (OUT direction).</li> </ul>
<b>Status stage</b>			
	Same as in <a href="#">Control OUT transfer examples</a> .		

### 36.4.3.5 Stream handling in SuperSpeed

The SuperSpeed stream protocol consists of a negotiation between the host and device concerning the selection of a stream, followed by data movement on the selected stream. The core automatically handles the stream protocol by initiating stream selection and also taking into account streams initiated by the host.

#### NOTE

This section is not applicable for USB 2.0 mode.

#### 36.4.3.5.1 Stream IDs and transfer resources

A stream ID is a 16-bit value which represents an individual flow of data between host and device. For stream-capable endpoints, the stream ID must be non-zero and is used in:

- The Command Parameter field in the DEPCMD register used when issuing a Start Transfer command.
- The Stream ID field in a TRB.
- The EventParam field in an endpoint-specific event (DEPEVT): XferComplete, XferInProgress, XferNotReady, and StreamEvt.

It is necessary to allocate 1 Transfer Resource per endpoint. The cost of each transfer resource is  $(32 + N\_TRBS\_PER\_XFER \times 16)$  bytes of memory in the descriptor cache. Core can cache maximum four TRBs per transfer.

After the Start Transfer command completes, the hardware returns the 7-bit Transfer Resource Index in the command complete event (and [Device Physical Endpoint-n Command Register \(USB\\_DEPCMDn\)](#)). This index is used only for the Update Transfer and End Transfer commands. In all other places, the Stream ID is used.

#### 36.4.3.5.2 Stream selection and stream programming model

Referring to the USB 3.0 Device Stream Protocol State Machine, the Idle state is the state where a device does not have a current stream to work on. When the hardware detects that it is in the Idle state, it will attempt to initiate and move data on a stream by transmitting ERDY to the host.

In order for this to occur, the endpoint must be "primed" and the stream must be both active and ready.

- **Primed:** After a stream-capable endpoint is configured, it is in the disabled state, which prevents the device from initiating any stream selection. After the host performs a Prime transaction, the endpoint is moved into the "Primed" state where it is eligible to initiate stream selection.

- **Active:** Software makes a stream active by issuing a Start Transfer command for the stream. A stream becomes inactive after an XferComplete event or after software issues an End Transfer command.
- **Ready:** After a stream is added via the Start Transfer command, its default state is "ready." The hardware will label the stream as "not ready" if the host gives a NoStream rejection to the hardware's attempt to initiate the stream. If the host performs a Prime, all currently active streams are automatically set to the "ready" state by hardware.

In the Idle state, the hardware selects a stream from its cache to initiate. Once the core has chosen an active stream, it attempts to initiate and move data on that stream until the endpoint enters the Idle state, which occurs when the host rejects the stream selection (NoStream) or the stream is terminated (PP=0) by the host or device. Therefore, software is required to prepare enough HWO=1 TRBs for at least one packet prior to issuing the StartXfer command.

In device-oriented stream selection class drivers (such as UASP), the host will not reject the device's stream selection. However, other class drivers may be developed in which the host initiates stream selection. In this situation, software will not start any transfers and will wait for a XferNotReady event. When that event is received, software will look at the StreamID in the event and start the transfer associated with the given Stream ID. If for some reason the class driver allows the host to reject a stream that has already been initiated by the host or device, software will receive a Stream (NotFound) event. If this occurs, software may replace the existing stream by issuing End Transfer for it and then issuing Start Transfer for another stream.

Some host implementation scenarios exist where the host and device become out of sync in the stream protocol, creating a deadlock condition where the device waits for the host to issue a Prime transaction while the host waits for the device to issue an ERDY. To resolve potential deadlock conditions, software should perform the following:

1. Implement a time out for stalled transfers on stream-capable endpoints.
2. When the timeout elapses with no data transfer for any stream after a StreamEvent (NotFound) event, restart one of the streams by issuing an End Transfer command followed by a Start Transfer command.

This places the stream in the Ready state and causes the core to transmit an ERDY to the host.

### 36.4.3.5.3 Data movement within a stream

Data movement within a stream is the same as data movement using the Bulk endpoint type (Start Transfer, Update Transfer, and updating TRBs). The XferInProgress, XferComplete, and XferNotReady events are issued as data is transferred between host

and device. The Stream ID field in the events indicate which stream within the endpoint is generating the events. If software receives an XferNotReady event with the EventStatus field indicating XferNotActive, it means that the host is attempting to initiate a stream that has not been added to the hardware's cache through Start Transfer. In response to this event, software should add the requested stream if it is available.

### **36.4.3.6 Low power operation**

#### **36.4.3.6.1 Low power operation of USB**

The USB 3.0 protocol allows a SuperSpeed host and device to negotiate with each other and decide when to bring their link into a lower power state (U1 or U2). This is accomplished by each side making individual decisions about whether they have any traffic currently pending or if they expect any traffic soon. The device requests low power entry by transmitting an LGO\_U1 (or LGO\_U2) Link Command and exits low power by using an LFPS handshake. The USB 3.0 Device Controller uses information from the host (bus side) and from software (application side) to decide when the best time to request low power entry is and when to exit from low power. The same calculation is used when the controller decides whether to accept a low power request from the host. The controller maintains state information about each endpoint and whether it is "active" or "paused". If all enabled endpoints are paused from either the application side or the bus side, low power entry is allowed. If at least one endpoint is not paused, low power entry is not allowed. To indicate that it does not have any pending traffic and allow the link to go into a low power state, the software may pause all enabled endpoints using one of the following mechanisms:

- Allow transfers to complete and not start any new ones
- Not setting the HWO bit in the TRBs to '1'

Exception: Software may prepare a Setup TRB for control endpoints without affecting the ability of the link going into low power. Otherwise, an endpoint is considered active on the application side. For these endpoints, the host indicates that it is paused in one of the following ways:

- Setting the Packet Pending (PP) flag to '0' in the DPH
- Setting the PP flag to '0' and NumP to '0' in the ACK TP
- For interrupt endpoints, by not polling the endpoint for two service intervals
- Note: After an endpoint is configured using the DEPCFG command and before the host sends any packets to it, the endpoint is considered to be paused on the bus side

Exception: If the data packet payload has a CRC error, the core responds with an ACK (Retry=1), and the endpoint is not paused. Otherwise, if PP=1 or Deferred=1, the endpoint is considered to be active on the bus side. Control endpoints have an additional condition as follows:

The core does not allow low power entry during a control transfer (starting from the reception of setup packet and ending at the completion of the status stage). If the core is configured for isochronous support, the following additional rules apply:

- The controller enters U0 at SystemExitLatency microseconds prior to the beginning of every microframe to receive the Isochronous Timestamp Packet (ITP).
- A PING causes every isochronous endpoint to wake up and be considered active, preventing low power entry.
- An isochronous endpoint is paused automatically when it transmits or receives the last packet of its interval (lpf=1) or if two intervals pass and the endpoint has not transmitted or received its last packet. In this way, after the PING, the core does not enter low power until all isochronous endpoints have moved their last packet or two intervals pass without another PING.

If all enabled endpoints are paused by software or the host, the core issues an LGO\_U1 (or LGO\_U2) link command or it responds with an LAU link command to the low power request of the host or hub. Otherwise, the core rejects the low power requests with an LXU link command.

Additionally,

- For bulk endpoints, the USB 3.0 specification says that the device may go to U1/U2 500ms (tERDYTimeout) after transmitting ERDY. It is the responsibility of the software to detect no activity on a bulk endpoint for 500ms or more and issue End Transfer if there is no activity. When the transfer is removed, the endpoint allows U1/U2 entry again.
- Because U1/U2 entry is based on the presence of packets in the TX FIFO and not based on the TRBs that are available to the core, it is possible that an IN endpoint responds with a NRDY, goes to U1, then U0, then transmits an ERDY if the timing and TX FIFO size works out such that the core cannot keep at least one packet in the TX FIFO at all times.
- In addition, before the final decision made to either initiate or accept low power state, device checks the state of DCTL[INITU2ENA][ACCEPTU2ENA][INITU1ENA] [ACCEPTU1ENA]. If the device receives Set Link Function LMP with Force\_LinkPM\_Accept bit asserted, the device accepts low power request from its link partner independent of endpoints state and DCTL Power control settings.

### 36.4.3.6.2 Low power operation of core

When the link is in the U1, U2, U3, SS.Inactive, SS.Rxdetect, or SS.Disabled state (for USB 3.0) or the USB is in Sleep, Suspend, or V<sub>BUS</sub>-off mode (for USB 2.0), the core can be put into a low power mode to save power. The core supports the following low power mode:

- **Clock-gating mode:** In this mode, the clocks connected to most of the core modules are gated off. Modules that still get clocks in the low power mode detect wakeup conditions and remove clock gating to other modules when a wakeup condition is detected. Clock-Gating mode can be used when the USB is in any low power state. Clock-Gating mode is enabled by setting the Disable Clock Gating bit in GCTL to 0.

#### 36.4.3.6.2.1 Clock-Gating Mode

Clock-Gating mode is enabled if the Disable Clock Gating bit in the GCTL is set to 0. When the USB/link is in a low power state, the clocks to most of the core modules are gated off automatically by the core. When a wakeup condition is detected by the core, or when the software schedules new work to the core (in U1 or U2 states) or changes the USB/Link state, the core turns on clocks and the entire core goes into active mode.

### 36.4.4 Host programming model

Refer xHCI specification for detailed information.

#### 36.4.4.1 Initializing host registers

In order to initialize the core as a host, the application should perform the steps described in the xHCI specification. Global registers need to be re-initialized as described in [Initializing global registers](#).

#### 36.4.4.2 Host controller capability registers

For register definition, refer to [Capability Registers Length and HC Interface Version Number \(USB\\_CAPLENGTH\)](#) to [Runtime Register Space Offset \(USB\\_RTSOFF\)](#).

#### 36.4.4.3 xHCI implementation details

This section discusses xHCI implementation details specific to the USB 3.0 controller.

### 36.4.4.3.1 LHCRST behavior

The xHCI Specification does not describe the programming model of light reset. It only specifies that the Operational and Runtime Registers that are not contained in the Aux Power well are at their default values.

Light reset must only be applied when USB\_DCTL[Run\_Stop] is equal to '0' to ensure that there is no discrepancy between hardware and software states. If light reset is applied during a transfer, the behavior on the USB is undefined, and may cause a packet to be terminated abruptly.

On light reset, the hardware resets all non-Aux registers to their default values which means that the port state (which is contained in PORTSC) does not change, but USB\_DCTL[Run\_Stop] is immediately set to '0' and all the context information about connected devices is lost. It is the responsibility of software to re-initialize the controller.

### 36.4.4.3.2 ENT requirements

In the xHCI Specification, the rules about when the ENT (Evaluate Next TRB) flag in a TRB must be set or cleared are not very strict. However, it states that the ENT flag must be set to '1' in the last Normal TRB when a TD ends with an Event Data TRB.

The controller requires software to follow these rules:

1. Regardless of whether the endpoint is stream-capable or not, if a TD ends with an Event Data TRB, the Normal TRB that precedes it must have ENT set to '1'. This Normal TRB may be separated from the Event Data TRB by a Link TRB
2. In all other cases, the ENT flag must be set to '0' Failure to follow these rules results in undefined behavior.

### 36.4.4.3.3 Behavior on babble error

If a babble error is detected and the received data passes all integrity checks, the host controller may write the received data (up to the expected data length) to the data buffer, and the value of the TRB Transfer Length field in the Babble Detected Error Transfer Event shall be consistent with the number of data bytes written to the buffer.

The controller does not write the received data to the data buffer. The entire packet is discarded and the residual byte count is written to the TRB Transfer Length field.

### 36.4.4.3.4 Max\_exit\_latency\_too\_large message

If periodic transfers are on every microframe (Binterval 1), the controller reports max\_exit\_latency\_too\_large if PING scheduling is enabled. The xHCI scheduler must prefetch data from the system memory ahead of next microframe, manage non-periodic transfer, and schedule PING before ISOC. Because the system memory access and USB responses are not predictable Quality of Service (QoS) is not guaranteed if the xHCI host needs to perform all of the above for every microframe when U1/U2 is enabled.

Therefore, the expectation is to disable U1/U2 if periodic data needs to be scheduled every microframe. In addition, because the U1 exit latency is in the order of 10usec, only very low system-level power saving is achieved even if the scheduler allows U1 transition by reducing QoS.

U1, U2, and U3 link states and exit time observed during lab testing:

- U1: Lowest power saving
  - In this state the chip is active, USB controller is inactive, only the Tx transmitter of the PHY is inactive.
  - U1 exit time per link partner: ~10 - 19 usec
- U2: Higher power saving than U1
  - In this state the chip is active, USB controller is inactive, both the Rx and Tx transceivers of the PHY are inactive.
  - U2 exit time per link partner: ~85 - 115 usec
- U3: Highest power saving
  - In this state, the chip and USB controller/PHY are inactive.
  - U3 exit time per link partner: 200 - 450 usec

Following are the recommendations for driver to handle periodic endpoints and "max\_exit\_latency\_too\_large" message:

- If Binterval of any endpoint is '1', then disable U1/U2 and do not schedule PING ("max\_exit\_latency" set to 0).
- If periodic scheduling is expected on every microframe, then disable U1/U2 and do not schedule PING.
- If periodic scheduling is expected only on every other microframe, then disable U2 and use U1 exit latency for "max\_exit\_latency" calculation.
- If  $((\text{number of hubs} + 1) * \text{U2 exit latency}) \geq (2^{**}(\text{Binterval}-1) * 125 \text{ usec})$ , then disable U2.
- For all other periodic scheduling use U2 exit latency for "max\_exit\_latency" calculation.
- If the "max\_exit\_latency\_too\_large" error is sent to software, then disable U2 and re-issue ep- config with U1 exit latency.
- If "max\_exit\_latency\_too\_large" still happens, then disable U1 and re-issue ep\_config with max\_exit\_latency set to 0 (no PING).

## 36.4.5 Device physical endpoint-specific commands

### 36.4.5.1 Command 1: Set endpoint configuration: DEPCFG

This command sets the physical endpoint configuration information.

#### NOTE

If GUSB2PHYCFG[6] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.

**Table 36-25. Command 1: Set endpoint configuration parameters: DEPCFG**

Field	Description
<b>Parameter 2</b>	
31-0	Set to endpoint state when the Config Action field of Parameter 0 is 1. Otherwise, this is Reserved.
<b>Parameter 1</b>	
31	FIFO-based Set to '1' if this isochronous endpoint represents a FIFO-based data stream where TRBs have fixed values and are never written back by the core.
30	Bulk-based Set to '1' if this isochronous endpoint represents a bulk data stream that ignores the relationship of bus time to the intervals programmed in TRBs.
29-25	USB Endpoint Number bit[29:26]: Endpoint number bit[25]: Endpoint direction: 0 OUT 1 IN Physical endpoint 0 (EP0) must be allocated for control endpoint 0 OUT. Physical endpoint 1 (EP1) must be allocated for control endpoint 0 IN.
24	Stream Capable (StrmCap) Indicates this endpoint is stream-capable (MaxStreams != 0).
23-16	blInterval_m1 Set to the blInterval value minus 1. The valid values for this field are 0 through 13. The blInterval value is reported in the endpoint descriptor. When the core is operating in Full-Speed mode, this field must be set to 0.
15	Set to '1' if this bulk endpoint utilizes the External Buffer Control (EBC) mode. This limits the number of outstanding DMA transfers to one read and one write. This is a requirement for the EBC to operate properly. Using EBC without setting this bit to '1' results in undefined behavior.
14	Reserved

*Table continues on the next page...*

## Functional Description

**Table 36-25. Command 1: Set endpoint configuration parameters: DEPCFG (continued)**

Field	Description
13-8	<p>Bits 13-8 are used for Device Endpoint Specific Event Enable (DEPEVTEN). If an enable bit is set to 0, the corresponding event is not generated.</p> <p>Bit 13 Stream Event Enable (StreamEvtEn)</p> <p>Bit 12 Reserved</p> <p>Bit 11 Reserved</p> <p>Bit 10 XferNotReady Enable (XferNRdyEn)</p> <p>Bit 9 XferInProgress Enable (XferInProgEn)</p> <p>Bit 8 XferComplete Enable (XferCmplEn)</p>
7-5	Reserved
4:0	<p>Interrupt number (IntrNum) Applies to IN and OUT endpoints.</p> <p>Indicates interrupt/Event Buffer number on which endpoint related interrupts for this endpoint are generated. This must be set to 0.</p>
<b>Parameter 0</b>	
31-30	<p>Config Action</p> <p>0 Initialize endpoint state: This action is used when an endpoint is configured the first time. It will cause the data sequence number and flow control state to be reset. DEPCMDPAR2 will be ignored. The encoding of this action is backward compatible with software that previously set "Ignore Sequence Number" to 0.</p> <p>1 Reserved.</p> <p>2 Modify endpoint state: This action is used when modifying an existing endpoint configuration, such as changing the DEPEVTEN event enable bits, interrupt number, or MaxPacketSize. The data sequence number and flow control state will be unchanged, and DEPCMDPAR2 will be ignored. The encoding of this action is backward compatible with software that previously set "Ignore Sequence Number" to 1.</p>
29-26	Reserved
25-22	<p>Burst Size (BrstSiz)</p> <p>When field is set to-</p> <p>0 Burst length = 1</p> <p>1 Burst length = 2, and so on, up to,</p> <p>15 Burst length = 16</p> <p>For IN transfers, this value represents the maximum length of the burst that the device attempts when the Host initiates an IN transfer with a TP_ACK.</p> <p>If BrstSiz &lt; the NumP value in the initiating TP_ACK, then the device controller limits the burst length to BrstSiz.</p> <p>If BrstSiz &gt;= the NumP value in the initiating TP_ACK, then the device controller attempts a burst length of NumP.</p> <p>For OUT transfers, this value represents the NumP value utilized in the response TP_ACK from the device controller. The NumP value indicates (to the host) the burst length the device desires. Note: In the special case of BrstSiz=0 (burst length = 1), the TP_ACK from the device controller has NumP=0; which is a flow-control condition. If this value is utilized, then the endpoint enters flow</p>

*Table continues on the next page...*

**Table 36-25. Command 1: Set endpoint configuration parameters: DEPCFG (continued)**

Field	Description
	control for each DP received and a subsequent ERDY is transmitted (according to the USB Specification). However, this may have an undesirable impact on the system throughput. To avoid this impact, it is recommended to set BrstSize=1 to prevent the flow-control condition. The trade-off is that the host could potentially burst two OUT data packets to the device, resulting in an ACK for the first packet, and an NRDY for the second packet.
21-17	FIFO Number (FIFONum) Indicates which transmit FIFO is assigned to this endpoint. For control endpoints, the FIFONum value in the OUT direction must be programmed to the same value as the IN direction. This field should be set to 0 for all other OUT endpoints. Even though there may be more than 16 TxFIFOs in DRD mode, the device mode must use lower 16 TxFIFOs.
16-14	Reserved
13-3	Maximum Packet Size (MPS) Applies to IN and OUT endpoints. The application must program this field with the maximum packet size (in bytes) for the current USB endpoint. USB 3.0 supports up to 1024 bytes.
2-1	Endpoint Type (EPType) This is the transfer type supported by this USB endpoint. 00 Control 01 Isochronous 10 Bulk 11 Interrupt
0	Reserved

### 36.4.5.2 Command 2: Set endpoint transfer resource configuration (DEPXFERCFG)

There must be only one transfer resource allocated per endpoint. Start transfer causes the use of the transfer resource. End Transfer or an XferComplete event releases the transfer resource. If software attempts to allocate more transfer resources than have been configured in the hardware, this command will return an error in the CmdStatus/EventStatus field.

#### NOTE

If GUSB2PHYCFG[SUSPENDUSB20] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.

**Table 36-26. Command 2: Set endpoint transfer resource configuration parameters: DEPXFERCFG**

Field	Description
<b>Parameter 2</b>	
31-0	Reserved
<b>Parameter 1</b>	
31-0	Reserved
<b>Parameter 0</b>	
31-16	Reserved
15-0	Number of Transfer Resources (NumXferRes) Defines the number of Transfer Resources allocated to this endpoint. This field must be set to 1.

### 36.4.5.3 Command 3: Get endpoint state (DEPGETSTATE)

This command is not used.

### 36.4.5.4 Commands 4 and 5: Set Stall and Clear Stall (DEPSSTALL, DEPCSTALL)

These commands apply to non-isochronous IN and OUT endpoints only.

The application issues a Set Stall command to stall all tokens from the USB host to this endpoint. If the endpoint is in the NRDY state, the STALL state takes priority.

If a transaction is currently in progress when the software issues Set Stall, the behavior depends on the direction of the endpoint:

- For OUT endpoints, the current packet will complete. The core will respond with STALL to the next OUT DP or token.
- For IN endpoints in Super Speed, the current burst transaction will complete, and the core will respond with STALL to the next ACK TP. For other speeds, the core will complete the current packet and respond STALL to the next IN token.

For non-control endpoints, the application is responsible for both setting and clearing STALL via the Set Stall/Clear Stall commands. When the application clears the STALL, the endpoint's data sequence number is reset to zero.

For control endpoints, the application issues only the Set Stall command, and only on the OUT direction of the control endpoint. The controller automatically clears the STALL when it receives a SETUP token for the endpoint.

The application must not issue the Clear Stall command on a control endpoint. If the endpoint is in flow control (NRDY) and also in the STALL state, it responds with a STALL for any packet. The only exception is that the controller always responds to SETUP data packets with an ACK handshake, independent of the STALL state.

### **NOTE**

- If GUSB2PHYCFG[6] is '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.
- When issuing Clear Stall command for IN endpoints in SuperSpeed mode, the software must set the "ClearPendIN" bit to '1' to clear any pending IN transactions, so that the device does not expect any ACK TP from the host for the data sent earlier.

#### **36.4.5.5 Command 6: Start transfer (DEPSTRXFER)**

This command applies to IN and OUT endpoints.

A Transfer Descriptor (TD) in Device mode is a list of TRBs that comprises one or more transfers. The pointer to a TD is added to the core's cache by the Start Transfer command and removed by an XferComplete event or by the End Transfer command.

Software issues this command indicating that a descriptor is ready to be processed and DMA can start for this endpoint/stream combination. For IN endpoints, this causes the descriptor to be processed, and data is moved from the corresponding memory buffer to corresponding transmit FIFO when the transfer is started. For OUT endpoints, this causes the descriptor to be processed and data is moved from the receive FIFO to corresponding memory buffer.

In response to the Start Transfer command, the hardware assigns this transfer a resource index number (XferRscIdx) and returns the index in the DEPCMDn register and in the Command Complete event. This index must be used in subsequent Update and End Transfer commands.

It is illegal to issue a Start Transfer command for the same TD if it is already present in the core's cache.

Non-stream capable endpoints rules:

- The Stream ID field must be set to 0.
- The HighPri field is reserved.

Stream-capable endpoint rules:

- The Stream ID field must be set to non-zero and match the Stream ID passed into the Start Transfer endpoint command associated with this transfer. In all the TRBs of a transfer, the Stream ID fields must be the same.
- The HighPri field is reserved.

### NOTE

- If GUSB2PHYCFG[6] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.
- Before issuing a start transfer command, software needs to verify whether the link is in U0. If the link is not in U0, software needs to bring the link to U0 by performing a remote wakeup. This is applicable to both SS and USB 2.0 speeds.
- When an IN transfer is in progress, (that is, started and not completed yet), and the software sees a link state change event to L1, then the software can use the GDBGIFOSPACE register (write followed by a read) to determine if the TxFIFO is empty or not. If the TxFIFO is not empty, it can initiate a remote wakeup.

**Table 36-27. Command 6: Start transfer parameters: DEPSTRTRXFER**

Field	Description
<b>Parameter 2</b>	
31-0	Reserved
<b>Parameter 1</b>	
31-0	Transfer Descriptor Address (TDAddr Low) Indicates the lower 32 bits of the external memory's start address for the transfer descriptor. Because TRBs must be aligned to a 16-byte boundary, the lower 4 bits of this address must be 0.
<b>Parameter 0</b>	
31-0	Transfer Descriptor Address (TDAddr High) Indicates the higher 32 bits of the external memory's start address for the transfer descriptor.

### 36.4.5.6 Command 7: Update transfer (DEPUPDXFER)

If software uses circular TRB buffers and updates a TRB, whose Hardware Owner (HWO) bit was 0, by setting HWO=1, it must execute the Update Transfer command, specifying the transfer resource index of the TRB in the DEPCMD register. The hardware uses this information to re-cache the TRB.

Software may issue a special “No Response Update Transfer” command by setting CmdAct=0 and CmdIOC=0. In this case, the hardware does not generate a Command Complete event, does not set the CmdAct bit to '0' (because it will be '0'), and software may immediately issue another command to the same endpoint following this one. This special type of Update Transfer may not be used when software depends on the XferNotReady event to setup TRBs (see “On Demand” transfers in [Transfer setup recommendations](#)).

Software may issue an Update Transfer command for a transfer resource that has already completed (either due to XferComplete event or an End Transfer command), and the core will detect that the Update Transfer is unnecessary. However, software must not issue an Update Transfer command for a transfer resource index that has never been started.

#### **NOTE**

If GUSB2PHYCFG[6] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.

#### **36.4.5.7 Command 8: End transfer (DEPENDXFER)**

This command applies to IN and OUT endpoints. Software issues this command requesting DMA to stop for the endpoint/stream specifying the transfer resource index of the TRB and the ForceRM parameter to be set to 1 in the DEPCMD register.

When issuing an End Transfer command, software must set the CmdIOC bit (field 8) so that an Endpoint Command Complete event is generated after the transfer ends. This is necessary to synchronize the conclusion of system bus traffic before the End Transfer command is completed.

For IN endpoints, this command causes descriptor processing to stop and the core stops fetching new data for the endpoint and stops transfers on the USB. The core may truncate a packet being transmitted with the DPPABORT ordered set, does not wait for any pending ACKs from the USB, and does not update the TRB status. For OUT endpoints, this command causes descriptor processing to stop. If there is currently data, it is moved from the receive FIFO to corresponding memory buffer and completed at packet boundary. The core does not update the TRB status.

Use this command under the following conditions:

- When handling USBReset or SetConfiguration, endpoints are closed using this command.
- After receiving a ClearFeature (STALL) control transfer, software issues End Transfer followed by Clear Stall, followed by Start Transfer.

- After an XferInProgress event when the TRB after the one that caused the XferInProgress event has its HWO bit set to '0', this command can be used.
- For isochronous endpoints, if the host stops moving data for many intervals, software may force the end of the transfer and wait for the host to restart.

The hardware does not issue a XferComplete event on End Transfer, but only issues a CommandComplete event.

### NOTE

If GUSB2PHYCFG[6] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.

**Table 36-28. Command 8: End Transfer Parameters: DEPENDXFER**

Field	Description
<b>Parameter 2</b>	
31-0	Reserved
<b>Parameter 1</b>	
31-0	Reserved
<b>Parameter 0</b>	
31-0	Reserved

### 36.4.5.8 Command 9: Start new configuration (DEPSTARTCFG)

Software issues this command under the following conditions:

- After power-on-reset with XferRscIdx=0 before starting to configure Endpoints 0 and 1. CmdIOC must be set to '0' and software must poll the CmdAct bit to determine when the command is complete because Endpoint 0 is not yet configured with a valid interrupt number.
- With XferRscIdx=2 when it receives SetConfiguration before starting to configure Endpoints > 1. CmdIOC may be set to '0' or '1'.

This command should always be issued to Endpoint 0 (DEPCMD0).

Hardware resets the transfer resource allocation to the value in the XferRscIdx parameter (must be 0 or 2) upon receiving this command.

### NOTE

If GUSB2PHYCFG[6] is set to '1', it must be set to '0' prior to issuing this command and may be set to '1' after the command completes.

## 36.4.6 OTG

This section describes OTG for USB 3.0, and the programming requirements for the USB 3.0 core in OTG mode.

### 36.4.6.1 OTG 2.0 for USB 3.0 Functionality

The following sections describe the OTG 2.0 functions for USB 3.0.

#### 36.4.6.1.1 Core OTG functions

The OTG Interface block within the U2MAC handles the core OTG functions: Session Request Protocol (SRP) and Host Negotiation Protocol (HNP). These OTG protocols are implemented through the regular UTMI+ OTG interface.

The MAC handles HNP for host and peripheral role swapping.

The MAC also handles SRP, which allows an A-Device to turn off VBUS to save power when the USB is not used, and provides a means for a B-Device to monitor VBUS and request the A-Device to activate VBUS.

It includes the following logic necessary to achieve SRP and HNP:

- Control of VBUS through Vbus drive enable (utmiotg\_drvvbus) as A-Device
- Control of IDDIG line sampling enable control output (utmiotg\_idpullup)
- Control of D+/D- pull-down resistor enables (utmiotg\_dppulldown/ utmiotg\_dmpulldown)
- Generates SRP (data line pulsing) as B-Device when the session is off.

#### NOTE

- When the application programs the USB\_GCTL[PRTCAPDIR] as 2'b11 to enable OTG mode, then in the A- Device mode, the core will only enumerate as a HS device and hence will not support any SS-capable device to be connected. This restriction is not valid when the USB\_GCTL[PRTCAPDIR] is programmed to 2'b01 or 2'b10.
- The utmisrp\_chrgvbus and utmisrp\_dischrgvbus outputs are provided for legacy PHY connections but they are both

- set to 1'b0 from the core since VBUS charging is not supported in OTG 2.0 specifications.
- The application should not program GCTL[0] as 1'b0 when it wants to do a ADP or HNP. In such a case, the application should disable the Clock Gating feature by programming GCTL[0] as 1'b1. The application should re-enable the Clock Gating feature only when the core returns to its original role of operation.

### 36.4.6.1.1.1 HNP Polling and Enable

This HNP Polling activity involves an OTG device acting as the current host to periodically poll the remote device to check if the remote device wishes to take the host role. It will then grant the role swap opportunity at the earliest opportunity. Being a software activity, HNP Polling is expected to be implemented through software timers and periodic exchange of SetFeature.SetHNPEnable packets.

The core is then informed of successful exchange of these packets to Enable HNP activity within the core.

### 36.4.6.1.2 ADP Functions

The ADP functions involve the following two processes:

- ADP sensing
- ADP probing

ADP probing capability is required in both A and B devices, while sensing is a must only for B-Device. The main functional unit is the ADP controller.

#### 36.4.6.1.2.1 Internal ADP controller

- All ADP timers are maintained in pwrm (Power) module and generate the enable and disable signaling to PHY for ADP probing and sensing.
- PHY contains the following circuitry related to ADP functionality:
  - Comparators for PRB and SNS
  - I\_ADP\_SRC and I\_ADP\_SNK
  - Vbus circuitry
- Application software programs the ADP timing registers residing in the pwrm module.
- pwrm module provides a mechanism to application through the interrupts to log and report events pertaining to ADP probing and sensing.

- Only pwrn module needs to be always powered on in this option. USB 3.0 controller may or may not be powered on.
- As a product, both the ADP controller logic and OTG controller are packaged into USB3 core. In the USB3 core, ADP sensing and probing is loosely coupled with the rest of the logic and is directly under software control. This achieves maximum flexibility to interact with OTG core functions.

The ADP controller in the block diagram is a fully digital controller. This has mainly timers inside it to help ADP operation. All ADP related timers are part of pwrn module.

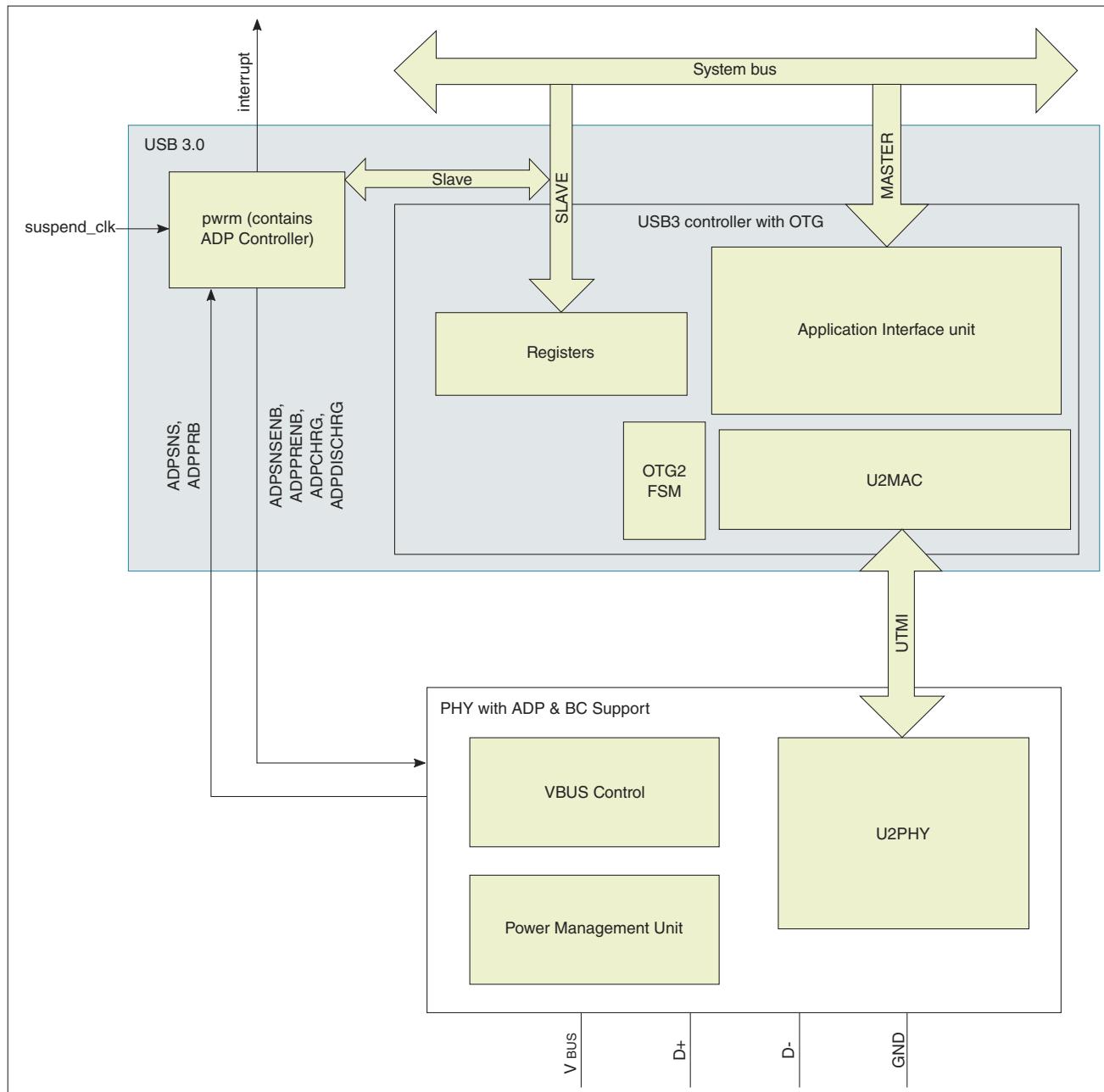


Figure 36-20. Internal ADP controller

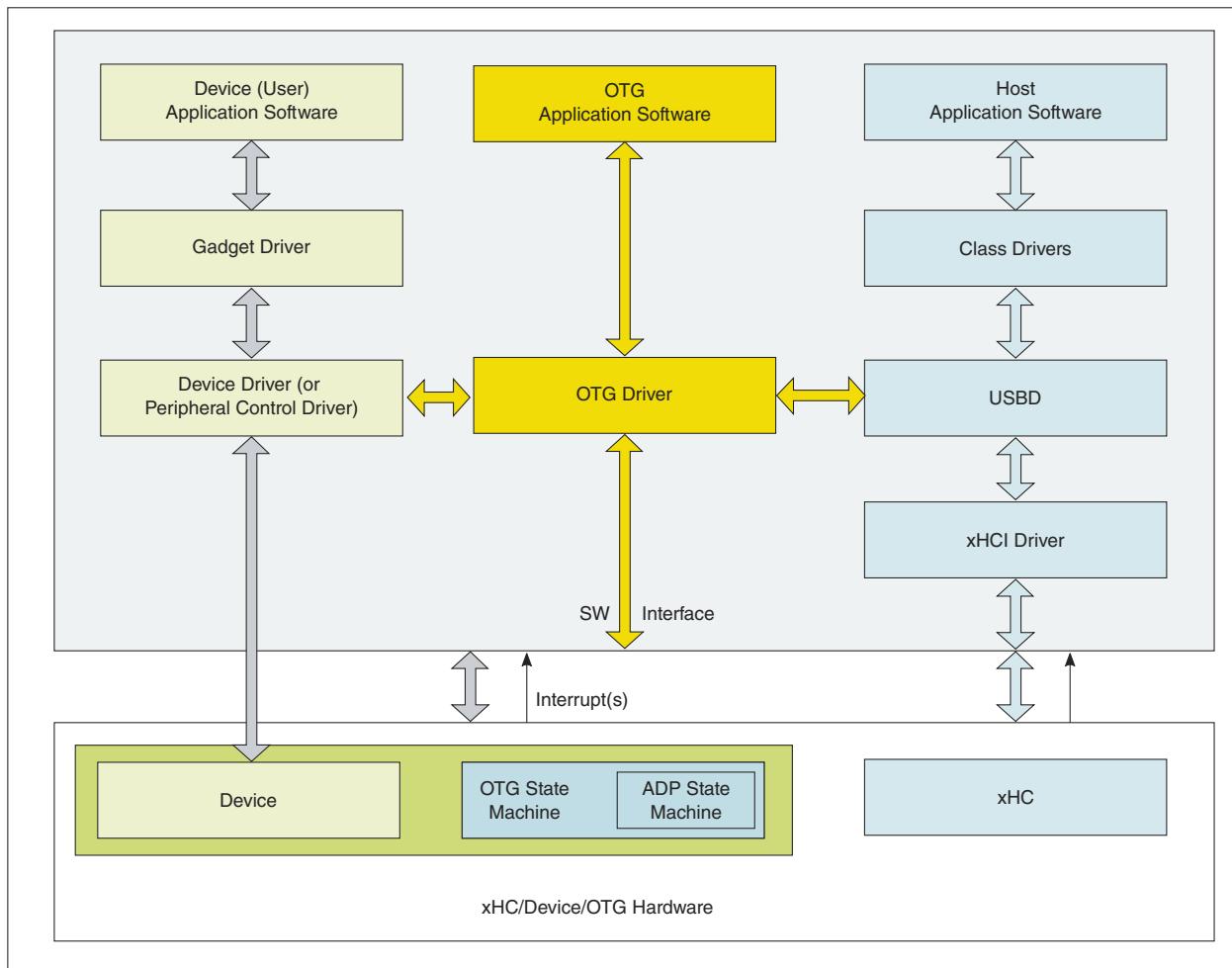
For the PHY to generate ADPPRB and ADPSNS, the PHY needs to know the value to compare with VADP\_PRB and VADP\_SNS respectively. These values can be hardcoded in the PHY or the PHY can have a limited programmable option by which these reference voltage values can be changed. It is assumed that the PHY takes care of this. The pwrm module also implements SRP detection in A-Device mode. This is required if pwrdwn module of the USB 3.0 controller is powered down completely while ADP is in progress and B-Device initiates SRP.

**NOTE**

- The ADP block works with the suspend clock.
- The ADP timers run with the suspend clock along with the programmed USB\_GCTL[PWRDNSCALE] value. The USB\_GCTL[PWRDNSCALE] value is used internally to generate 32 KHz pulse for the timers.

#### **36.4.6.1.3 Software flow**

The software flow diagram is illustrated in the following figure.



**Figure 36-21. Software flow diagram**

The following two sections describe the A-device and B-device flows briefly with respect to the above flow diagram.

### NOTE

The following concise flows are not very detailed, and are provided here to help understand at an abstract level.

#### 36.4.6.1.3.1 A-Device activity concise flow

- Power-on reset
- Only the OTG driver is active after power-on reset. The Host and Device drivers are inactive after power-on reset since at this point of time it is unsure what role the device assumes.
- The OTG driver initializes ADP and waits until a successful ADP event is received.
- The OTG driver senses ID pin = 0 and initializes the OTG registers.
- The OTG driver waits until a successful SRP event is received and turns on the VBUS.

- The OTG driver requests the USBD (USB driver) to bring up the xHC.
- After successful xHC bring up, device enumeration (and data transfers) occurs during which the device capabilities are exchanged for determining a possible role switch.
- In the event where a role switch is possible, the OTG application software or the B-device may initiate a role switch (The OTG application software may request via the Host application software to create favorable conditions for a role switch).
- The xHC is shut down and the control is passed on to the OTG driver.
- The OTG driver requests the PCD (Peripheral driver) to bring up the Peripheral.
- After successful Peripheral bring-up, device enumeration (and data transfers) occurs.
- In an event where a role switch is possible, OTG application software may initiate a role switch (The OTG application software may request via the Device application software for a role switch).
- The PCD is shut down and the control is passed on to the OTG driver.
- The OTG driver requests the USBD to bring-up the xHC, thus reverting to the original roles.

#### **36.4.6.1.3.2 B-Device activity concise flow**

- Power-on reset
- Only the OTG driver is active after power-on reset. The Host and Device drivers are inactive after power-on reset since at this point of time it is unsure what role the device assumes.
- The OTG driver initializes ADP and waits until a successful ADP event is received.
- The OTG driver senses ID pin = 1 and initializes the OTG registers.
- The OTG driver initiates a SRP and waits for the VBUS to be turned on by the A-device.
- The OTG driver requests the PCD to bring up the Peripheral.
- After successful Peripheral bring up, device enumeration (and data transfers) occurs during which the device capabilities are exchanged for determining a possible role switch.
- In an event where a role switch is possible, the OTG application software or the A-device may initiate a role switch (The OTG application software may request via the Device application software for a role switch).
- The PCD is shut down and the control is passed on to the OTG driver.
- The OTG driver requests the USBD to bring up the xHC.
- After successful xHC bring-up, device enumeration (and data transfers) occurs.
- In an event where a role switch is possible, the OTG application software or A-device may initiate a role switch (The OTG application software may request the Host application software for a role switch).
- The xHC is shut down and the control is passed on to the OTG driver.
- The OTG driver requests the PCD to bring-up the Peripheral, thus reverting to the original roles.

### 36.4.6.1.4 Programming model

This section describes the programming requirements for the USB 3.0 core in OTG mode. The core can be configured either as a DRD-device or an OTG, based on bit [13:12] of the Global Control Register's (GCTL) Power-On Initialization Value. If power-on configuration option is chosen as DRD-device, then PrtCapDir has to be explicitly programmed to 2'b11 for operating as an OTG device. The premise for the programming model is as follows.

2. The OTG driver is expected to run concurrently and independently with the PCD and xHCI drivers.
3. The OTG driver doesn't get involved in the actual data transfers but is responsible for communicating messages between the xHCI driver and PCD during role changes.
4. The OTG driver is responsible for handling the HNP, SRP and ADP (internal) events from the core. After power on, the OTG driver is responsible for loading, starting and switching between the xHCI driver and PCD. When started, the xHCI driver initializes the xHCI register set before enumerating the connected device. Similarly, when started, the PCD initializes the device register set and waits for the USB reset event to continue. Once the connection is established between the xHCI driver or the PCD by the OTG driver, the corresponding driver takes over for data transfers.

After every successful HNP switching, it is necessary that the active driver is changed from the xHCI driver to PCD or vice-versa. In such cases, the application may also unload the active driver from the memory and re-use the same memory area for loading the next active driver. The subsequent flow described in this section assumes that the core can get enumerated in any of the speeds (HS/FS) and bases its discussion accordingly. For example, if the core enumerates in HS/FS, follow the OTG 2.0 flow.

#### 36.4.6.1.4.1 Initializing global registers

The global registers of the USB 3.0 core are shared between the device and host modes. It is the responsibility of the individual driver to initialize the core's global register specific to its functions. This is not explicitly mentioned in the flows described in this chapter. This specifically includes initializing the GTXTHRCFG/GRXTHRCFG, GEVNTEN, and GPRTBIMAP registers, and the Global FIFO Size and Global Event Buffer Registers. For more details, refer to Global registers in [Table 36-2](#). However, these global registers that are programmed once at power on can be initialized in the global initialization routine in the OTG driver. Examples of one-time programmable registers are GSBUSCFG, GCTL, GSNPSID, and GUCTL. Examples of fields that are one-time programmable are the PHY interface (UTMI), AXI parameters like burst size, and so on.

The standard xHCI host driver does not get involved in the core-specific global register programming, and therefore, needs to be handled by the Board Support Package software.

#### **36.4.6.1.4.2 Initializing host registers**

In order to initialize the core as a host, the application should perform the steps described in the xHCI specification. Global registers need to be re-initialized as described in [Initializing global registers](#).

#### **36.4.6.1.4.3 Initializing device registers**

In order to initialize the core as a device, the application should perform the steps described in the section [Device Programming Model](#). Global registers need to be re-initialized as described in [Initializing global registers](#). For specific registers, refer to [Device Programming Model](#).

#### **36.4.6.1.4.4 Initializing OTG registers**

The application should initialize the OTG registers based on the initial value of the OSTS.ConIDSts after power on. The following sections depict the programming flow for the OTG application in detail.

#### **36.4.6.1.4.5 Programming flow for OTG in USB 3.0**

The following figure shows the programming flow after power-on reset.

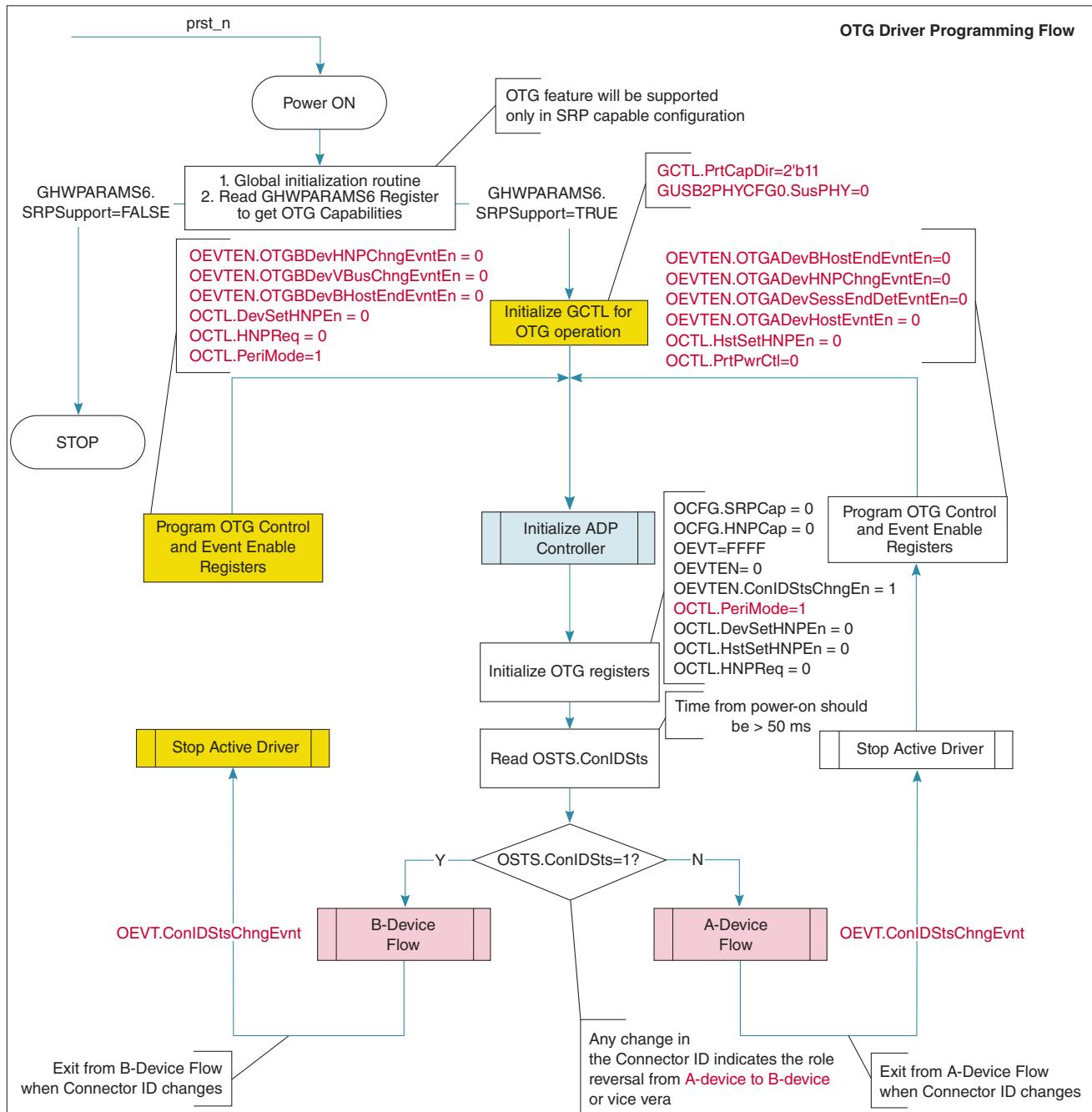


Figure 36-22. OTG driver overall programming flow

Following steps describe overall programming flow:

1. During global initialization, read the GHWPARAMS6 register to see if SRP support is enabled in the configured core. If SRP support is enabled, then proceed with the OTG programming flow by programming USB\_GCTL[PRTCAPDIR] as 2'b11. If SRP support is not enabled, do not proceed further in the OTG programming flow. The configured device must at least support SRP for cable connection-based (Connector ID) role of operation.

2. Initialize the ADP controller as explained in ADP programming flow.
3. The OTG 2.0 state machine is initially in B-IDLE if IDDIG is 1, or A-IDLE if IDDIG is 0. Enable USB\_OEVTE[ConIDStsChngEvntEn] for any change in the Connector ID status.
4. Initialize the OTG control and configuration register to default values.
5. Read the OSTS register to find out the current status of the Connector ID (ConIDSts). After power-on or soft reset, the ConIDSts will be valid only after the PHY delay from IDPULL=1 to IDDIG active and any filter delay for IDDIG inside or outside the USB 3.0 core.
6. If USB\_OSTS[ConIDSts] is 1, the OTG 2.0 state machine is in B-IDLE, follow the B-Device flow. Otherwise, if OTG 2.0 state machine is in A-IDLE, follow the A-Device flow.
7. When there is any connector ID (IDDIG) change resulting in USB\_OEVT[ConIDStsChngEvnt], then exit the A-Device/B-Device flow. Stop the active driver and re-initialize the OTG control and status registers.

#### **36.4.6.1.5 Common driver tasks**

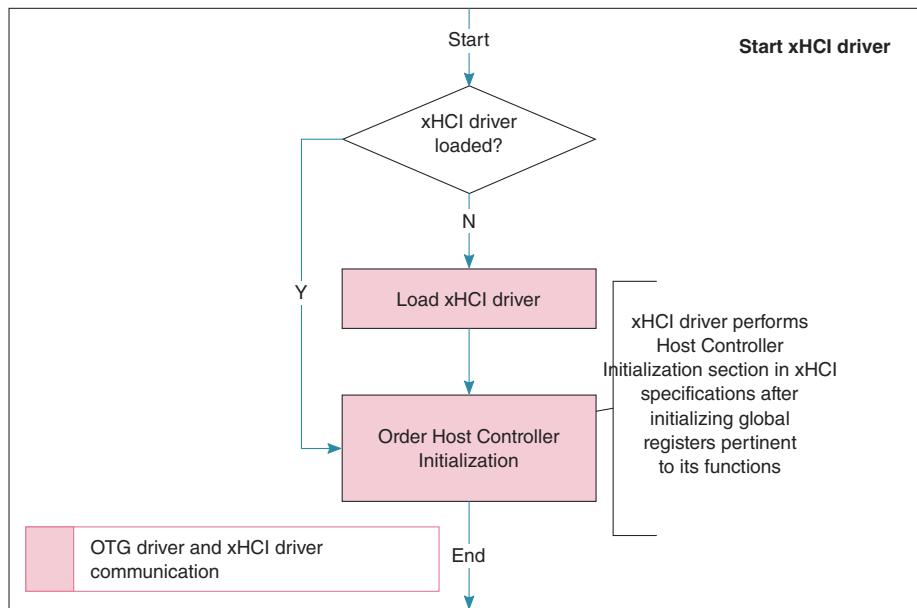
The common tasks for both A-Device and B-Device flow are as follows:

- Start xHCI driver
- Start peripheral control driver (PCD)
- Switch peripheral
- Switch host
- Stop active driver
- Enable HNP change

The Stop Active Driver is already introduced in the common flow diagram. Other tasks are useful for the A-Device/B-Device flows.

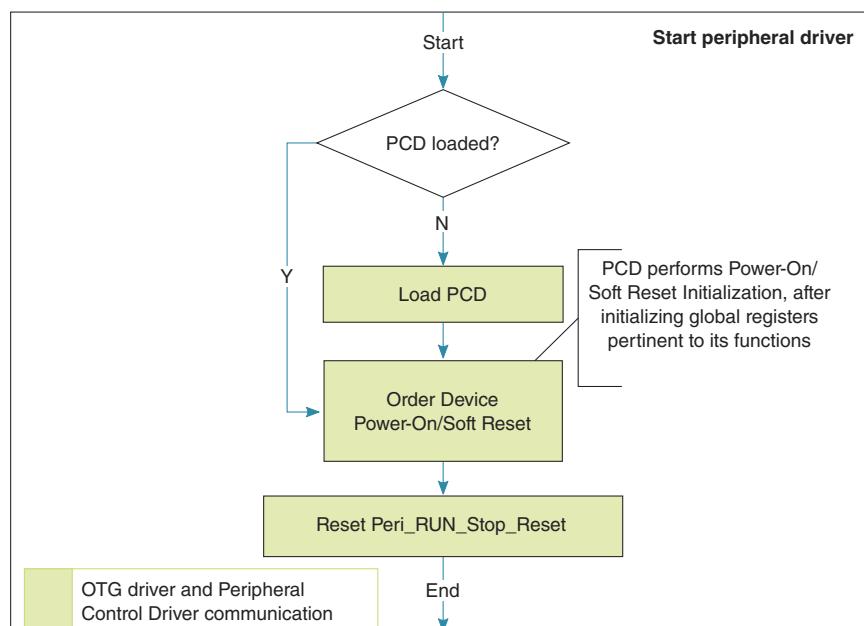
#### **Start xHCI driver**

The OTG driver, being responsible for starting the xHCI driver, executes this task. The OTG driver loads the xHCI driver after initializing the global registers, and the xHCI driver is responsible for all the functions thereafter.

**Figure 36-23. Start xHCI driver task**

### Start peripheral control driver

The OTG driver, being responsible for starting the Peripheral Control Driver (PCD), executes this task. The OTG driver only loads the PCD, which is responsible for all the functions thereafter, including the global registers initialization specific to the USB 3.0 core. Note that except the GTXFIFOSIZ and GEVNTADR, none of the other global registers need to be potentially re-programmed with different values.

**Figure 36-24. Start peripheral driver task**

## **Stop active driver**

This task checks which driver is currently active and stops the active driver. Optionally, it may unload the driver if the OS supports dynamic loading of drivers.

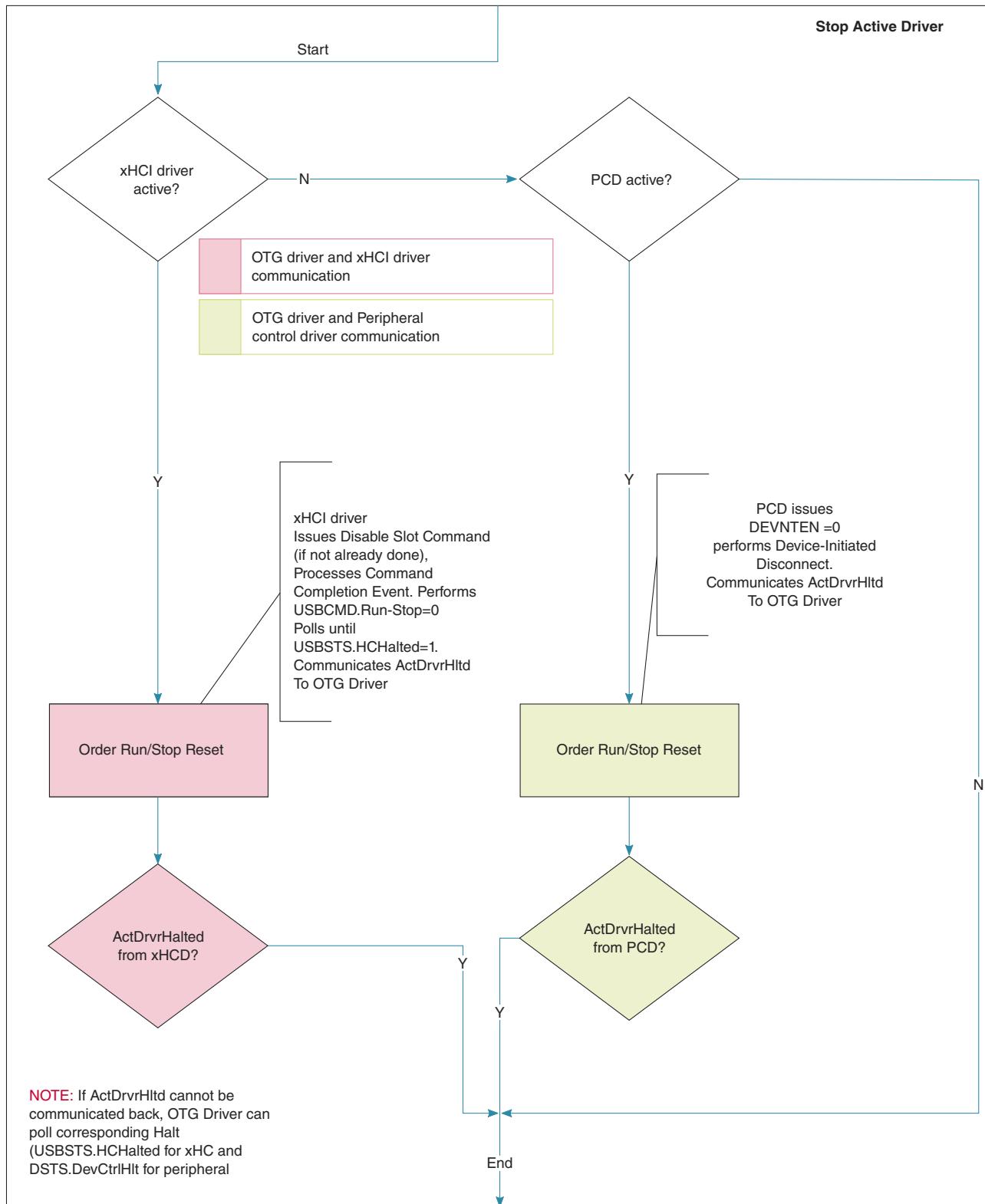
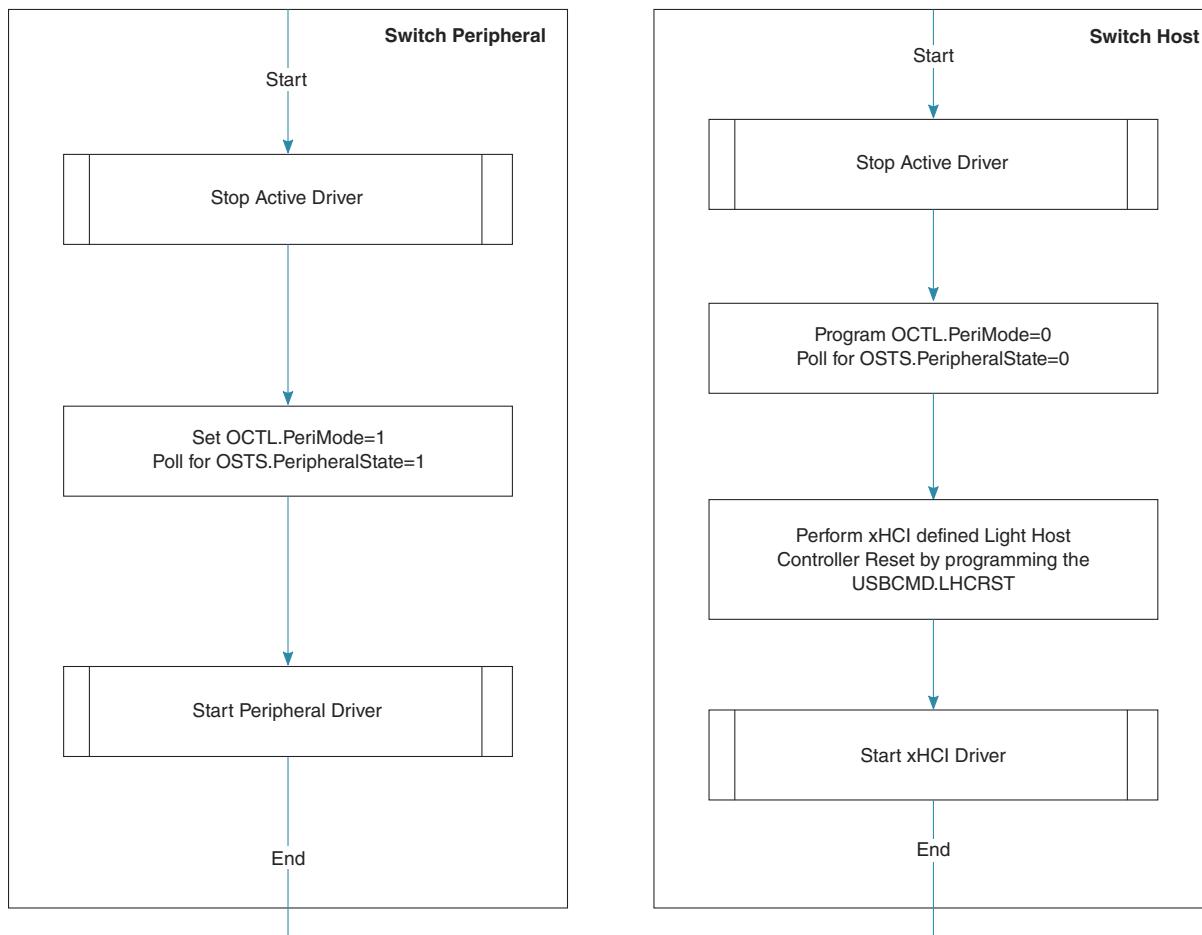


Figure 36-25. Stop active driver task

**Switch peripheral and switch host**

These tasks are used during HNP and each consists of two individual tasks.



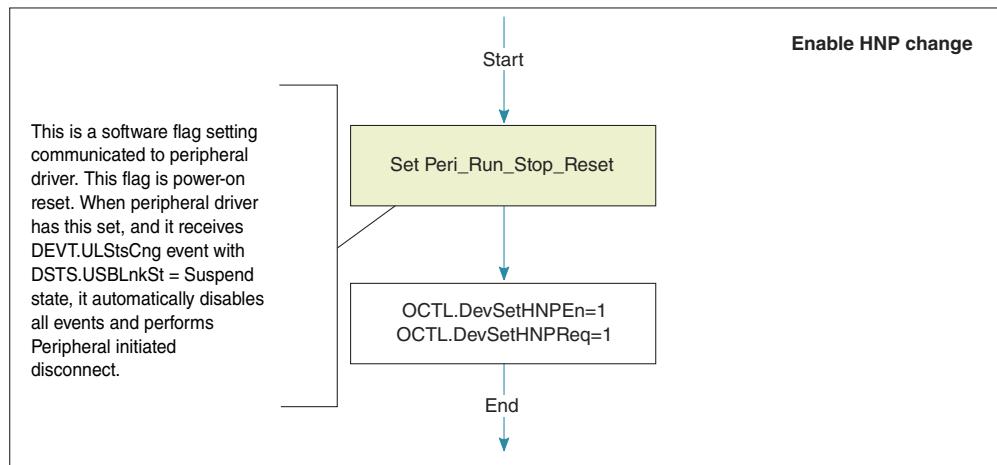
**Figure 36-26. Switch peripheral task and switch host task**

### Enable HNP change

For HS/FS, this task is called when SetFeature.SetHNP is successfully exchanged between the host and the device. The set Peri\_Run\_Stop\_Reset indicates to the device driver to automatically disable all events and perform a peripheral-initiated disconnect when the host puts the device in Suspend. Program the OCTL.DevSetHNPEn as 1, and OCTL.DevSetHNPReq as 1 immediately after the successful completion of SetFeature.SetHNP on the USB. This ensures the HNP attempt from the OTG core in the following Suspend state.

#### NOTE

By default, at power-on, program GUSB2PHYCFG.SusPHY as 1 to ensure that the PHY enters low power mode during normal suspend/resume without role switch.

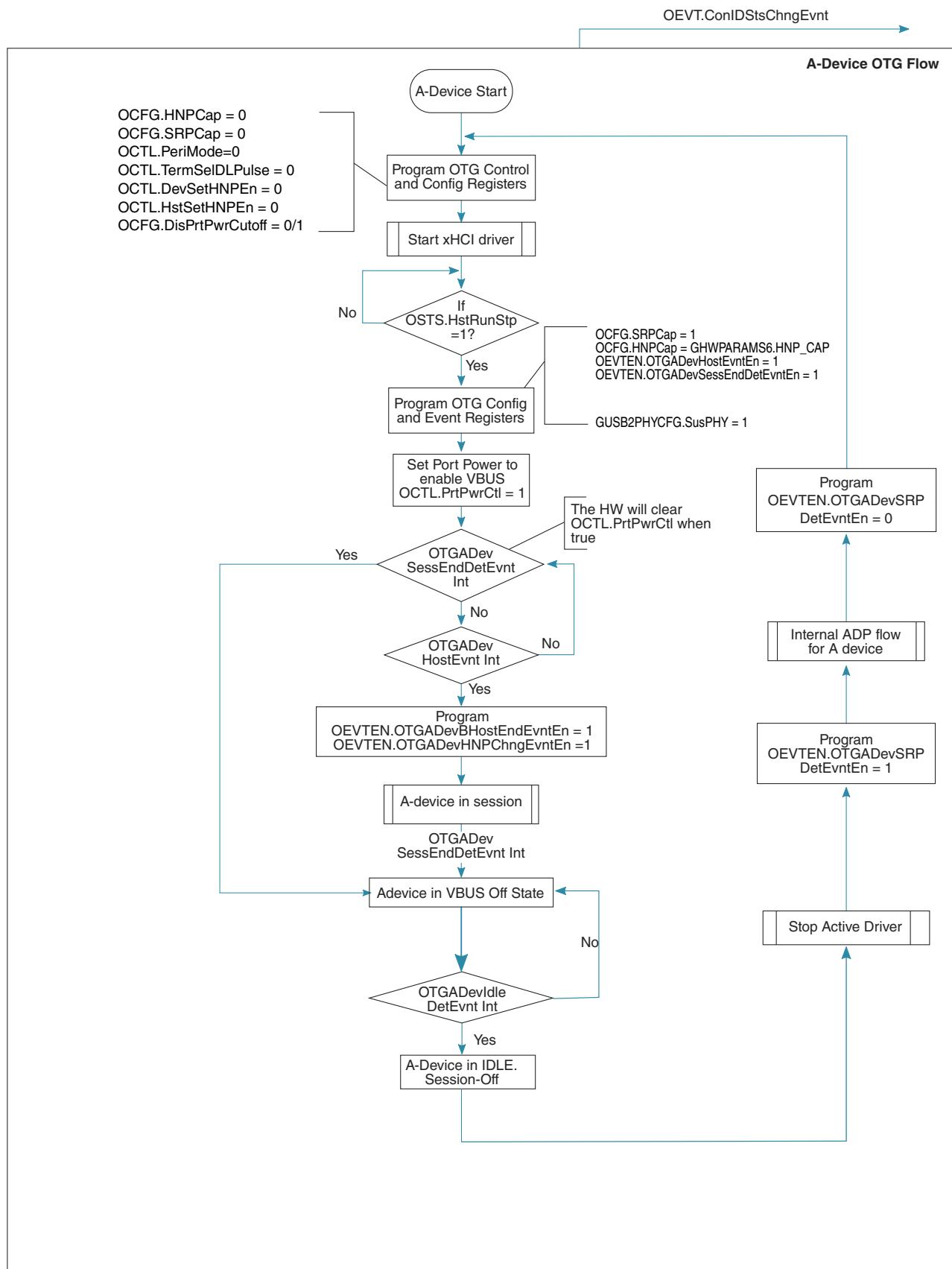


**Figure 36-27. Enable HNP change**

#### 36.4.6.1.6 A-Device flow

The following figure shows the A-Device flow.

## Functional Description



**Figure 36-28. A-Device flow diagram**

**A-Device flow:**

1. At Start, the A-Device core will be in A\_IDLE VBUS off state.
2. The OTG Software should program the OTG control and configuration registers for A-Device operation:
  - OCFG.HNPCap = 0
  - OCFG.SRPCap = 0
  - OCTL.PeriMode = 0
  - OCTL.TermSelDLPPulse = 0
  - OCTL.DevHNPEn = 0
  - OCTL.HstSetHNPEn = 0
  - OCFG.DisPrtPwrCutoff = 0 or 1, based on whether the application requires the core disable the feature to automatically switch off VBUS after 1 second if Port is disconnected.
3. Start the xHCI driver.
4. The OTG Software should wait for OSTS.HstRunStp to become 1 to continue. If OSTS.HstRunStp is 0, the OTG Software should wait for a software event that indicates the change in the xHCI Run/Stop bit to continue. Alternatively, the OTG Software can poll for OSTS.HstRunStp bit to be set.
5. OTG Software should program the OTG Configuration and Event registers to enable A-Device events to detect peripheral connect and session end. Refer to A-Device Flow Diagram.
6. Program GUSB2PHYCFG0.SusPHY=1 to enable the core to switch off the PHY clock in the possible non-HNP Suspend scenarios.
7. Set OCTL.PrtPwrCtl when OSTS.xHCIPrtPower is 1. This enables driving of VBUS to the B-Device.
8. The A-device may enumerate in HS/FS based on the device signaling.
9. The OTG 2.0 state machine enters a3\_ds\_host/A-host for HS (FS) if there is a successful B-device connect (ADevHostEvnt).
10. The OTG software should enable the ADevHNPCChng and ADevBHostEnd events to detect the transition from A-Host to A-Peripheral and the end of A-Peripheral events.
11. If there is an ADevSessEndDetEvnt interrupt indicating that the core has now stopped driving the VBUS, the software should wait for ADevIdleDetEvnt Interrupt and then disable all the OTG events.
12. The Software can enable the ADP flow.
13. After an ADevHostEvnt interrupt, the core is in session. The flow for the A-Device in session is shown in earlier in this chapter.

**NOTE**

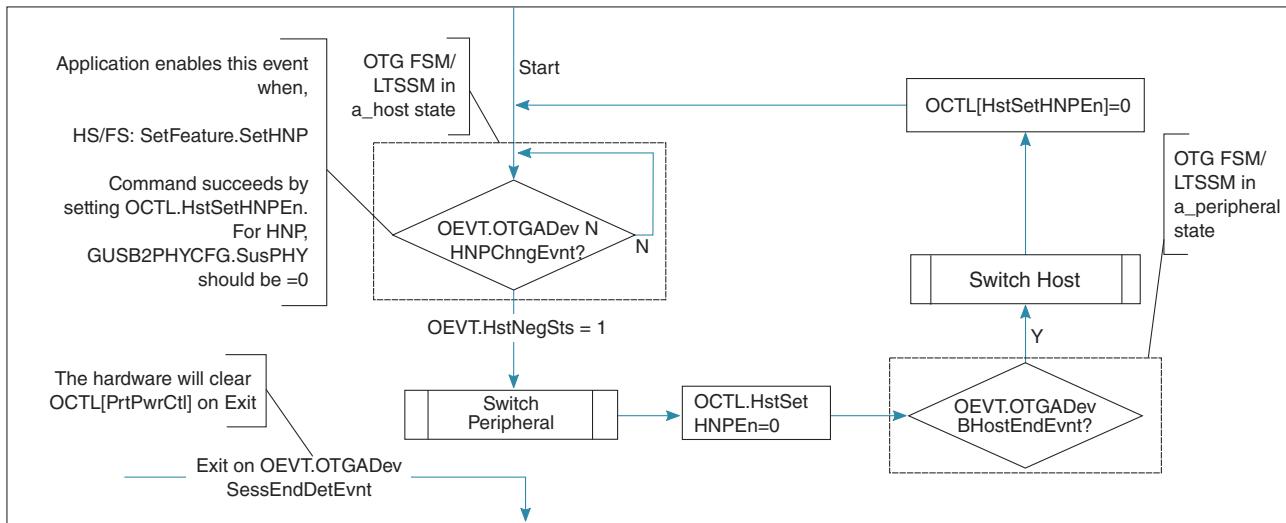
- At any step, if ConIDStsChngEvnt occurs, A-Device flow is terminated as per the figure "OTG Driver Overall Programming Flow".
- Over current is not detected by the OTG state machine. The xHCI controller detects it, and the OTG driver in turn sees an ADevSessEndDetEvnt (from PORTSC.PP = 0) due to an overcurrent.

**OTG state machine mapping to the software flow**

- Steps 1-5 are A-IDLE.
- Step 7 is A\_WAIT\_VRISE in which A-Device has detected SRP and therefore waits for VBUS to stabilize.
- Step 8 and step 10 are primarily A\_HOST through A\_WAIT\_VRISE and A\_WAIT\_BCON. If there is an error in either A\_WAIT\_VRISE or A\_WAIT\_BCON, ADevSessEndDetEvnt occurs making the state-machine fall back to A-IDLE.

**Handling Over-Current in OTG mode of operation**

- When the core is operating in USB 2.0 mode, an over-current condition causes PORTSC.OCA=1 and PORTSC.OCC=1 on the corresponding USB 2.0 port number. The core also updates a Port Status Change Event TRB for the corresponding USB 2.0 port number.
- When the core is operating in USB 3.0 mode, an over-current condition causes PORTSC.OCA=1 and PORTSC.OCC=1 on the corresponding USB 2.0 and USB 3.0 port numbers. The core also updates a Port Status Change Event TRB, one each for the corresponding USB 2.0 port number and the corresponding USB 3.0 port number.
- Note that PORTSC.OCA may transition quickly from 0 to 1 and 1 to 0 before xHCI driver can see two distinct interrupts. This may occur due to over-current condition being removed quickly. In such a case, the xHCI driver will receive a single interrupt (PORTSC.OCC) for over-current.

**Figure 36-29. A-Host -> A-Peripheral -> A-Host Flow Diagram****For HS/FS mode:**

- When the core starts the session with A-Device as host, the application may periodically exchange SetFeature.SetHNPEnable packets. If B-Device wants to become the host, this packet exchange results in the host application knowing about B-Device's intent to become the host, and it directs the OTG driver to set OCTL.HstSetHNPEn. This enables the core to cater for HNP from B-Device when suspend is initiated. During suspend, ADevHNPCChngEvnt is set.
- If ADevHNPCChngEvnt is set, the OTG driver reads OEVT.HstNegSts. If OEVT.HstNegSts is set, then the core transitions to A-Peripheral state by executing Switch Peripheral task. Otherwise, wait for ADevSessEndDetEvt and go to ADP probing.
- When in A-Peripheral state, if there is ADevBHostEndEvnt, it signifies that the B-Device no longer wants to be the host. Therefore, the driver goes back to Step 1 by executing the Switch Host task and assumes an A-Host role.  
Alternatively, if there is an xHCI.PrtPower de-assertion, the state machine goes to start ADP probing due to the occurrence of ADevSessEndDetEvt.
- If ADevBHostEndEvnt is set, then the core would be in A-Host state by executing Switch Host task. Otherwise, perform either of the following:
  - Drop VBUS and go to ADP probing.  
If the role change is successful, the OTG driver hands over the control to the xHCI driver.

**Internal ADP flow for A-Device**

The internal ADP flow of A-Device is as follows:

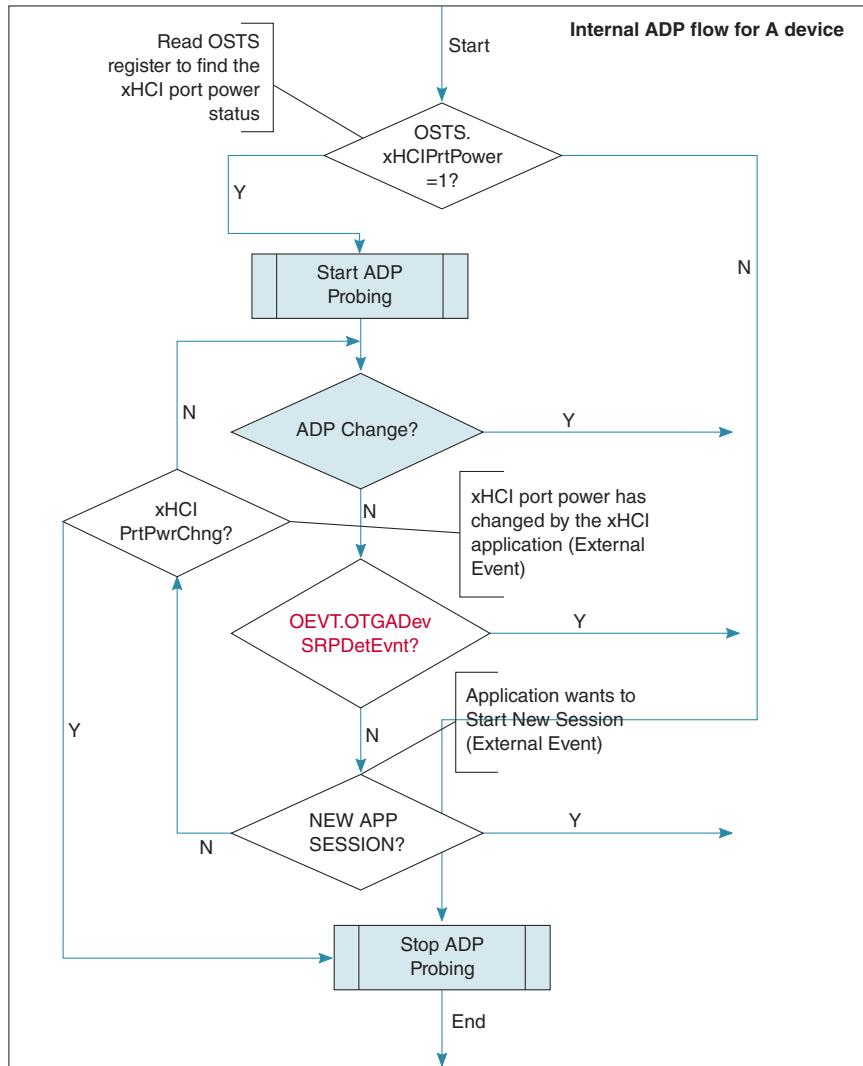
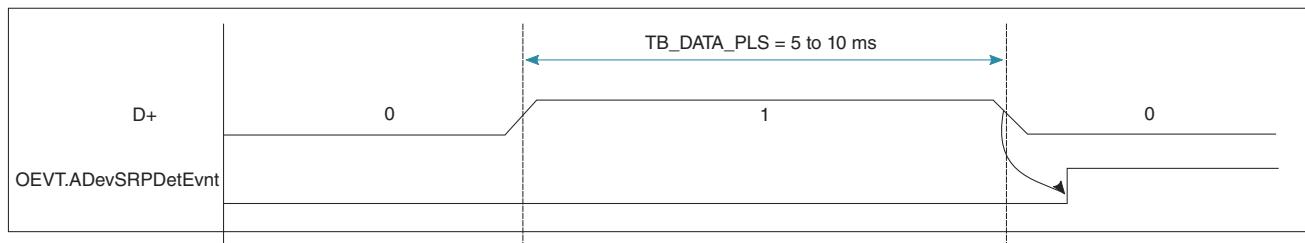


Figure 36-30. Internal ADP flow for A-Device

- ADevSessEndDetEvnt occurs due to:
  - PORTSC.PrtPower = 1'b0
  - OTG driver stopping VBUS due to:
    - BCON\_TOUT when in A\_WAIT\_BCON
    - A\_AIDL\_BDIS\_TOUT when in A\_SUSPEND
- At any step, if ADevSessEndDetEvnt occurs, the software starts with ADP probing.
- Once the ADP probing (both external and internal) is started, the software should wait for one of the following events to occur (to stop ADP probing and proceed to Step 2 in A-Device flow) to continue.
  - ADP change event
  - SRP Detect event
  - New Session event from A-Device application
  - xHCI port power change event from A-Device application

### 36.4.6.1.7 SRP detection by the core (Timeline for ADevSRPDetEvnt)

ADevSRPDetEvnt is triggered by a valid DLINE pulsing on the USB D+ line by B-Device. The A-Device looks at both the positive and negative edge transition occurrence for a valid data-line pulse time before detecting SRP.



**Figure 36-31. Timeline for ADevSRPDetEvnt**

### 36.4.6.1.8 VBUS turned ON by the core (Timeline for ADevBSessEndEvnt)

When the A-device turns on the VBUS, it is unsure at this point of time at what speed the USB enumerates to. So, both the LTSSM and the OTG 2.0 FSMs start concurrently. The LTSSM proceeds with the training sequence after Rx\_detect and the OTG 2.0 FSM waits for A\_WAIT\_BCON.

**HS/FS Mode:** Before entering A-Host, there is a check done in the OTG state machine to ensure B-Device is connected. If B-Device is not connected within TA\_WAIT\_BCON, ADevBSessEndDetEvnt is set. The core resets OCTL.PrtPwrCtl, and the OTG state machine enters A\_WAIT\_VFALL. Then VBUS to B-Device is removed and this results in ADevSessEndEvnt. Note that the A\_WAIT\_VFALL transition can occur due to over current, which is not depicted in the following figure.

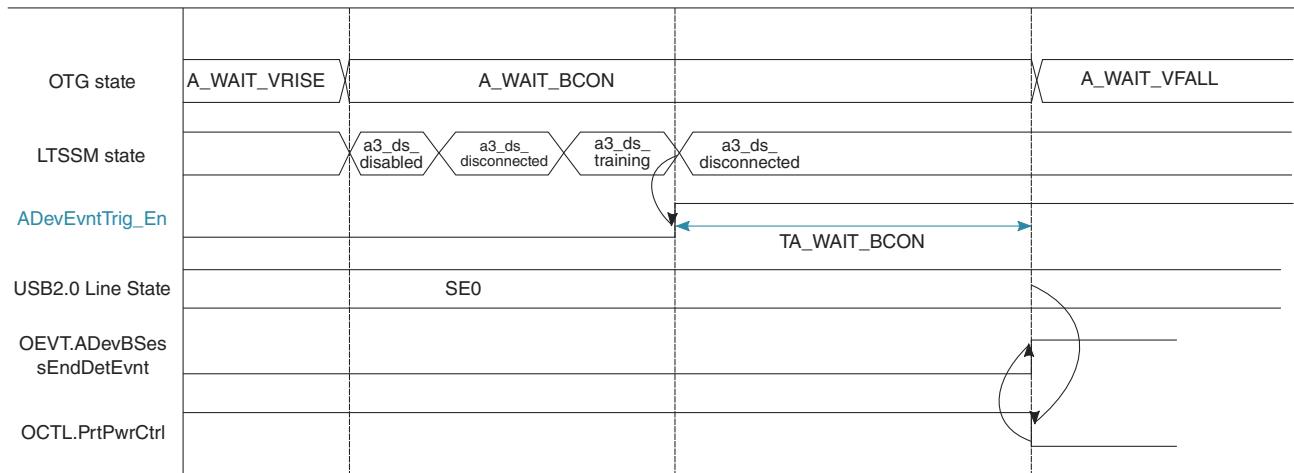


Figure 36-32. Timeline for ADevBSessEndEvnt

#### 36.4.6.1.9 Core entering A-Host in HS/FS mode (Timeline for ADevBHostEvnt)

**HS/FS Mode:** If the B-Device is connected within TA\_WAIT\_BCON, ADevBHostEvnt is set.

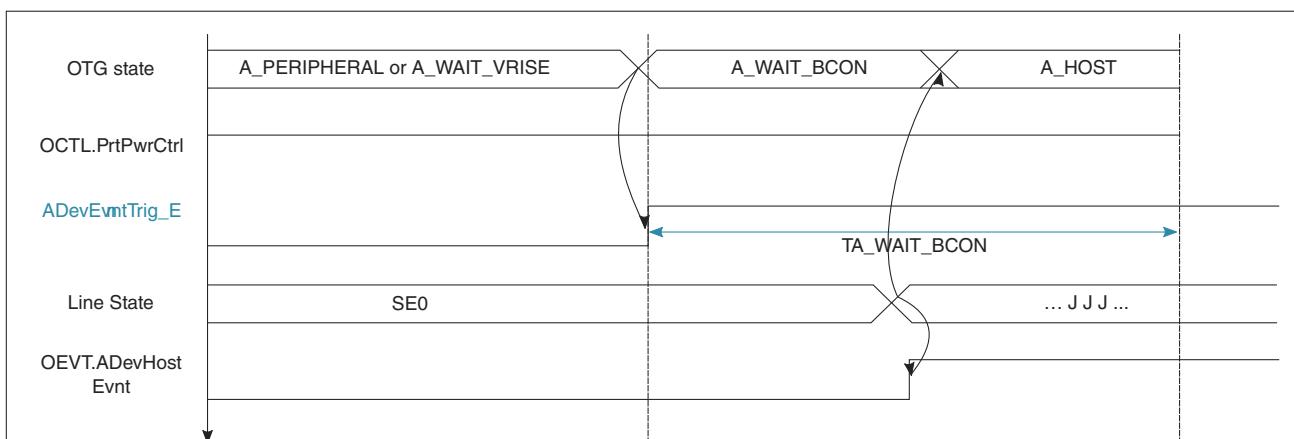


Figure 36-33. Timeline for ADevBHostEvnt

#### 36.4.6.1.10 B-Device flow

The following figure shows the B-Device flow.

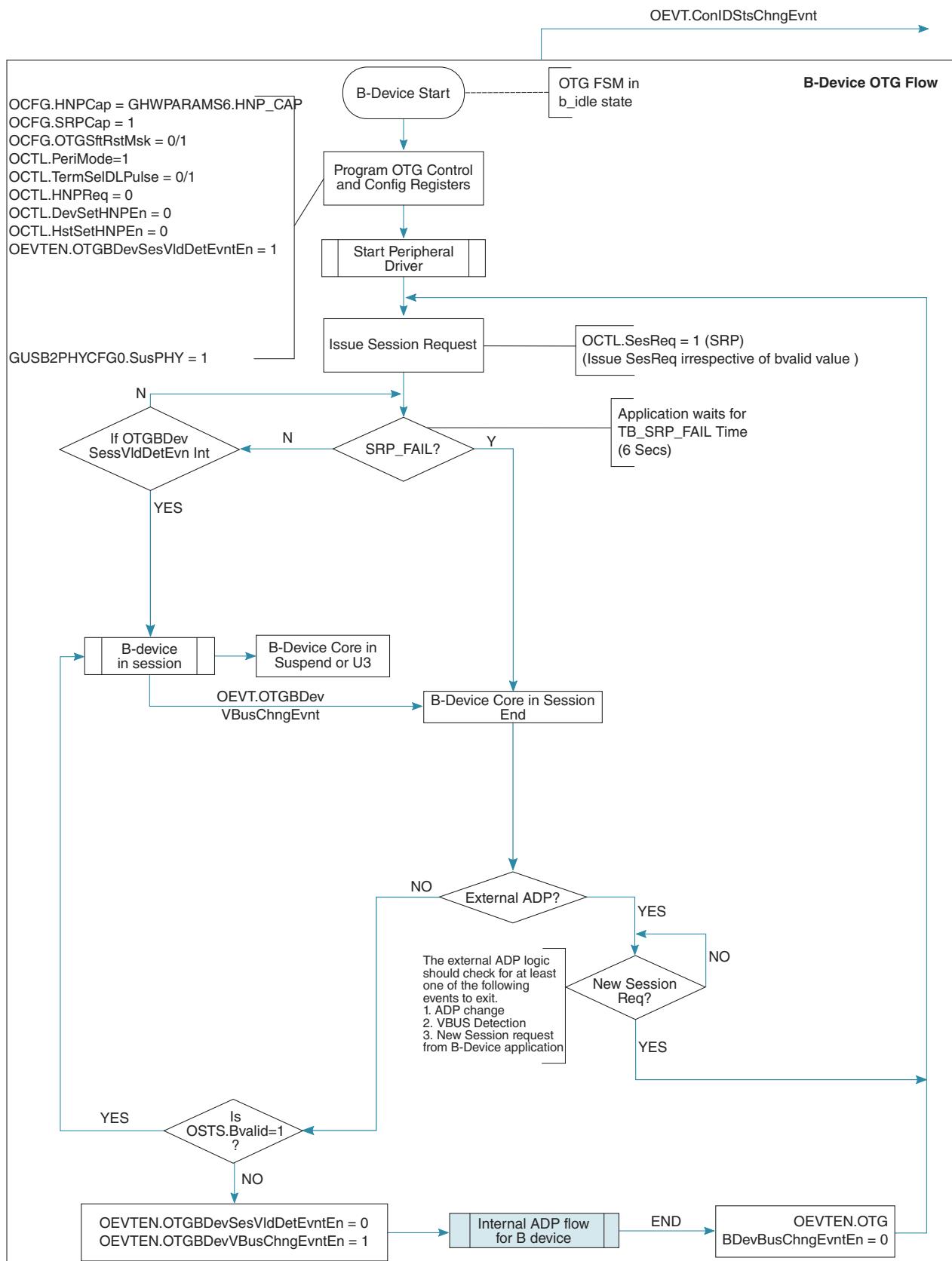


Figure 36-34. B-Device flow diagram

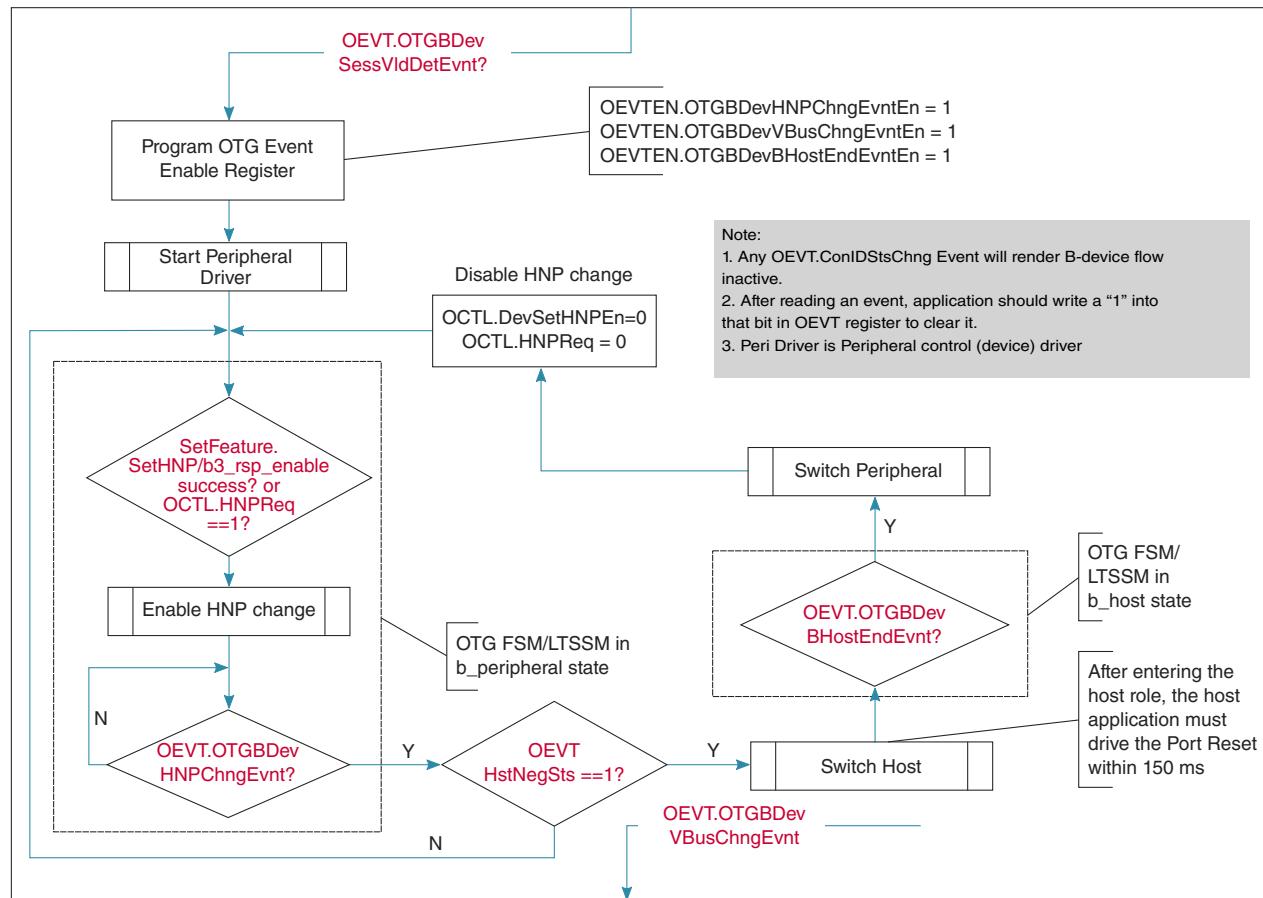
1. At Start, the core will be in B-IDLE state. Note that for OTG functionality in Peripheral mode, DCTL.Run/Stop should be set, otherwise the core as B-Peripheral will not initiate SRP pulsing. Also, it is important for the Device Driver to program the speed of operation as B-peripheral (DCFG.Spd).
2. The OTG Software should program the following OTG Control Registers for B-Device operation:
  - OCFG.HNPCap = GHWPARAMS6.HNP\_CAP
  - OCFG.SRPCap = 1
  - OCTL.PeriMode = 1
  - OCTL.TermSelDLPulse = 0
  - OCTL.HNPReq = 0
  - OCTL.DevHNPEn = 0
  - OCTL.HstSetHNPEn = 0
  - OEV滕.OTGBDevSesVldEvntEn = 1
  - GUSB2PHYCFG0.SusPHY = 1
3. Start Peripheral Driver.
4. Program OCTL.SesReq=1 to initiate SRP by D-line pulsing. The core initiates SRP and waits for a valid VBUS (bvalid).
5. The B-Device core continues to wait for BVALID if the A-Device does not respond. The application should timeout if SessVldDetEvnt does not come within the SRP fail time (TB\_SRP\_FAIL is 6 seconds).
6. When SRP fails, the application should enable ADP probing .
7. If the A-Device asserts a valid VBUS in response to SRP, the core reports an BDevSessVldDetEvnt to indicate the start of a session.
8. If there is a BDevSessVldDetEvnt, the core enters a session on state. For HNP Flows in B-Peripheral -> B-Host -> B-Peripheral flow diagram.
9. If there is a BDevVBusChngEvnt anytime, indicating VBUS is no longer valid, the core enters Session- End state.

### **NOTE**

- For OTG 2.0, during HNP process where the B-Device is going to assume the role of a host, the B-Device application needs to ensure that a USB reset process is programmed within 150 ms (TB\_ACON\_BSE0) of getting a device connect interrupt.
- At any step, if ConIDStsChngEvnt occurs, the A-Device flow is terminated as per "A-Device Flow Diagram".

OTG state machine mapping to the software flow is as follows:

- Steps 1-3 and 6 are B-IDLE.
- Step 4 is B-SRP-INIT.
- Step 7 and 8 are B-PERIPHERAL.



**Figure 36-35. B-Peripheral -> B-Host -> B-Peripheral flow diagram**

### For HS/FS mode:

1. The OTG state machine starts operation as a B-peripheral. The following events are enabled.
  - `OEVTE.N.OTGBDevHNPChngEvntEn = 1`
  - `OEVTE.N.OTGBDevVBusChngEvntEn = 1`
  - `OEVTE.N.OTGBDevBHostEndEvntEn = 1`

In this state, the application periodically exchanges `SetFeature`.`SetHNPEnable` packets. If the B- Device wants to become host, this packet exchange results in a device application sending an ACK to this packet, and it directs the OTG driver to execute Enable HNP Change. Ensure that `GUSB2PHYCFG0.SusPHY` is `1'b0` when `SetFeature`.`SetHNP` command succeeds. This enables the core to initiate HNP from B-Device when a suspend is initiated. During a suspend, `BDevHNPChngEvnt` is set. `OTGBDevHNPChngEvnt` starts the transition from B-Peripheral ->B-WAIT-ACON to either B- Host or B-Peripheral.

2. If `BDevHNPChngEvnt` is set, the OTG driver reads `OEVT.HstNegSts`. If `OEVT.HstNegSts` is set, it indicates successful connection of A-Device as peripheral,

the core therefore transitions to B-Host state. After entering the host role, the host application must drive the port reset within 150 ms. If OEVT.HstNegSts is not set, it indicates failure of HNP and it stays as B-Peripheral.

3. When in B-Host state, if there is BDevBHostEndEvnt, it signifies that the B-Device no longer wants to be the host, therefore suspended the bus and A-Device has stopped signaling connect. Therefore, the driver goes back to step 1 and assumes B-peripheral role.
4. The core generates BDevBHostEndEvnt.

**NOTE**

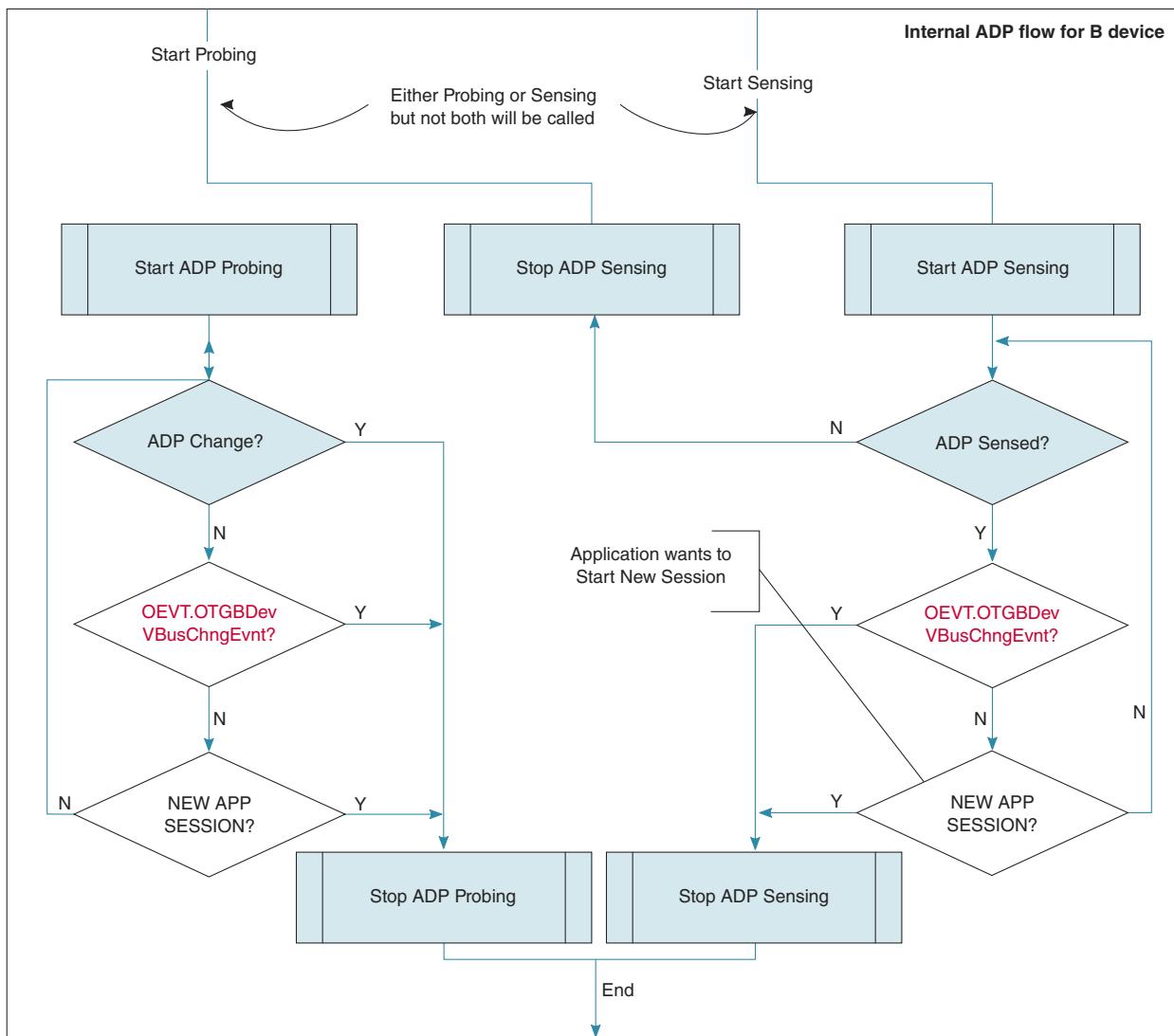
- Step 3 above is B-Host and B-Host to B-PERIPHERAL.
- OTGBDevVBusChngEvnt indicates transition back to B-IDLE.

**NOTE**

- Step 3 above is B-Host and B-Host to B-Peripheral.
- OTGBDevVBusChngEvnt indicates transition back to B-IDLE.
- If there is any unexpected error scenario like VBUS drop after the OCTL.DevSetHnpEn bit is set, the OTG driver must clear the OCTL.DevSetHNPEn bit.

**Internal ADP flow for B-Device**

The internal ADP flow of B-Device is as follows:

**Figure 36-36. Internal ADP flow for B-Device**

While probing, the application should wait for one of the following events to exit probing and start SRP request again.

- ADP Change
- VBus detection
- New session request from B-Device application

While sensing, the application should wait for one of the following events to exit sensing.

- ADP sensed failed
- VBus detection
- New session request from B-Device application

The application should continue with ADP probing if ADP sense failed. Else, the application should exit sensing and start SRP request again

### 36.4.6.1.11 Core entering b3\_us\_peripheral in B-Peripheral in HS/FS mode (Timeline for BDevSessVldDetEvnt)

The PCD decides the speed in which the USB3 core needs to operate by programming DCFG.DevSpd bits. Accordingly, the core decides to operate in FS to start with.

HS/FS Mode: BDevSessVldDetEvnt is triggered by a valid VBUS detection by B-Device in B-IDLE state. The B-Device will transition to B-PERIPHERAL state before triggering this event. Note that this is common to both SS and HS(FS).

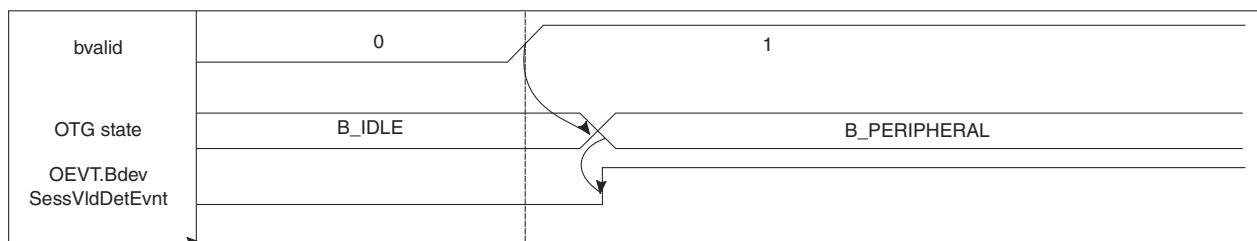


Figure 36-37. Timeline for BDevSessVldDetEvnt in HS/FS mode

### 36.4.6.1.12 VBUS change detected on USB (Timeline for BDevVBusChngEvnt)

BDevVBusChngEvnt is triggered when a change in bvalid is detected irrespective of OTG state machine.

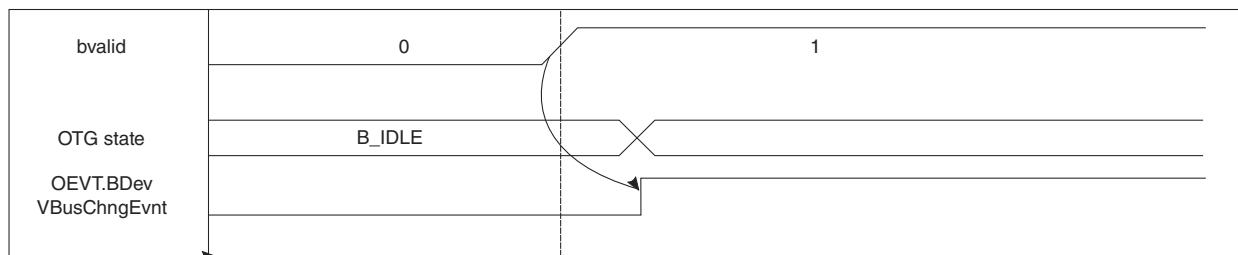


Figure 36-38. Timeline for BDevVBusChngEvnt

### 36.4.6.1.13 Internal ADP controller logic

The ADP controller has timers that assist in ADP operation. The ADP-related registers are a part of the pwrm\_otgif module. For the PHY to generate ADPPRB and ADPSNS, the PHY needs to know the value to compare with VADP\_PRB and VADP\_SNS, respectively. These values can be hardcoded in the PHY or the PHY can have limited programmable option by which these reference voltage values can be changed. The

"Wakeup Logic" module implements SRP detection in A-Device mode. This is required if the USB 3.0 controller is powered down completely while ADP is still in progress and B-Device initiates SRP.

### **NOTE**

ADP controller logic is inferred along with, but outside USB 3.0 controller.

- As a product, both ADP controller logic and OTG controller are packaged into USB 3.0 core.
- All ADP timers are maintained in the ADP controller module, which generates the enable and disable signaling to the PHY for ADP probing and sensing.
- PHY contains the following circuitry related to ADP functionality:
  - Comparators for PRB and SNS
  - I<sub>\_</sub>ADP\_SRC and I<sub>\_</sub>ADP\_SNK
  - V<sub>BUS</sub> circuitry
- Application software programs the ADP timing registers that resides in the pwrn\_otgif module.
- The pwrn\_otgif module provides a mechanism to the application via interrupts to log and report events pertaining to ADP probing and sensing.
- In this operation, only the pwrn module needs to be always powered on. The pwrwn module can be either powered on or off.

### **36.4.7 Initialization/application information**

The PHY can be configured for fixed equalization by programming relevant control registers in the USB 3.0 PHY.

In order to initialize the USB 3.0 PHY, software should perform the following steps:

1. Write 1'b0 to RX\_OVRD\_IN\_HI.RX\_EQ\_EN [address 16'h1006: bit 6].
2. Write 1'b1 to RX\_OVRD\_IN\_HI.RX\_EQ\_EN\_OVRD [address 16'h1006: bit 7].
3. Write a fixed value to RX\_OVRD\_IN\_HI.RX\_EQ [address 16'h1006: bits 10–8] [equalization setting] (generally 2–4, based on testing in customer environment).
4. Write 1'b1 to RX\_OVRD\_IN\_HI.RX\_EQ\_OVRD [address 16'h1006: bit 11].

## 36.4.8 Power management overview

The following power management features are available in the USB 3.0 core:

- Hardware-controlled LPM in USB 2.0 host
- Clock gating in device (U1/U2/U3) and host (U1/U2/U3)

The power saving mechanism is hardware-controlled LPM. This is a USB 2.0 host-only feature where the host controller automatically detects the downstream port is idle for a certain amount of time and autonomously initiates an LPM token to the connected device. If the device accepts, the link is put into the standard USB 2.0 Suspend state.

Finally, clock gating is also supported in the host controller as well as in the device controller. The following table gives the list of power saving mechanisms available in USB 3.0 core.

**Table 36-29. Power savings support**

USB 3.0 State	USB 2.0 State	Device Power Savings	Host Power Savings
U1 (Hardware Initiated)	None	Core: Clock Gating PHY: P1	Core: Clock Gating PHY: P1
U2 (Hardware Initiated)	LPM-L1 (Hardware Initiated)	Core: Clock Gating, PHY: P2/Sleep	Core: Clock Gating, PHY: P2/ Sleep
U3 (Software Initiated)	Suspend (Software Initiated)	Core: Clock Gating, PHY: P3/Suspend	Core: Clock Gating, PHY: P3/ Suspend
Rx.Detect (Software Initiated)	None	None	Core: Clock Gating, PHY: P3/ Suspend
SS.Disabled (Software Initiated)	None	Core: Clock Gating, PHY: P3/Suspend	None

### 36.4.8.1 Clock gating

The core implements clock gating in the following situations:

- In USB 2.0 device mode
  - When the UTMI suspend or l1\_suspend signal is asserted and the core is idle
- In USB 3.0 device mode, when the link is in U1, U2 or U3 state and the core is idle.

In host mode, clock gating is enabled in the following situations:

- When all USB 2.0 ports are in the suspend or L1 Suspend state, and all USB 3.0 ports are in the U1/U2/U3 state, and the core is idle.

Internal clock gating will apply to:

- Modules that use the ram\_clk
- Some modules that use bus\_clk

Internal clock gating will not apply to:

- Modules that use mac3\_clk: When the USB 3.0 PHY is suspended, these modules switch to using suspend\_clk, so their power consumption is reduced but the clock is not completely stopped.
- Modules that use mac2\_clk: When the USB 2.0 PHY is suspended, these modules switch to using suspend\_clk, so their power consumption is reduced but the clock is not completely stopped.
- The bus\_gs module, which uses bus\_clk. This module detects wakeup on the slave interface and thus needs a non-gated clock.

Clock gating can be enabled or disabled using the GCTL register. The GCTL register is "sticky," it retains its value across soft resets.

### 36.4.8.2 Hardware-Controlled LPM

In USB 2.0, the main link power management mechanism was for the host to suspend the link. However, suspending the link requires a long period of idle and was wasteful of power, especially because the host knows ahead of time whether it is going to schedule any traffic. It also required a long period of resume signaling.

The USB 2.0 Link Power Management Addendum [LPM-ECN] added a new token to the USB 2.0 specification, the LPM token. The host has the option of suspending the link quickly instead of waiting for a long period of idle, and communicates its desire to the device by sending an LPM token. The ECN also introduced a terminology for different link states. This new link state was called L1. The traditional USB 2.0 suspend was called L2. The L1 state requires a much shorter resume signaling period too.

An xHCI driver can initiate LPM by writing '2' to the PORTSC.PLS field. However, there is a lot of overhead for an xHCI driver to be constantly trying to keep the link in a low power state. Because of this, the xHCI specification also allows a compatible xHC to be enabled with "Hardware-Controlled LPM" by setting PORTPMSC.HLE as 1. When this happens, the xHC is expected to automatically send LPM tokens when it has no pending transfers and thinks that the link is idle enough to enter L1. This feature is supported by the host core.

### 36.4.8.2.1 Special consideration for OTG

When the core is configured as an OTG device, it will not initiate LPM when acting in a reverse role as a B- Host. This is because suspend on the link is treated as a request to switch back to its original role.

# Chapter 37

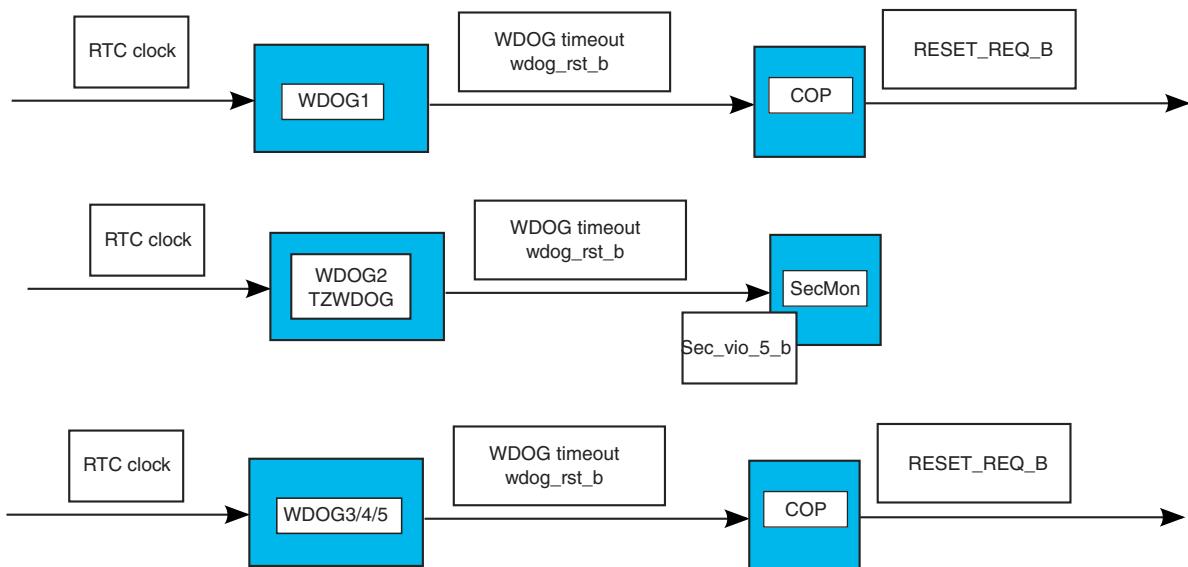
## Watchdog Timer (WDOG)

### 37.1 Overview

The Watchdog Timer (WDOG) protects against system failures by providing a method by which to escape from unexpected events or programming errors.

The chip has five WDOG timers. Once the WDOG is activated, it must be serviced by the software on a periodic basis. If servicing does not take place, the timer times out. Upon timeout, the WDOG asserts the timeout signal (wdog\_rst\_b) as reset request to COP.

Following figure shows system level representation of watchdog.



**Figure 37-1. WDOG system level diagram**

One WDOG is dedicated for Trustzone support, other four are for each core(1 for each core in the cluster). The Trustzone WDOG expiry signal is used to raise a security violation to the Security Monitor, the core WDOG timers expiry signals are used to raise reset request (RESET\_REQ output of the chip).

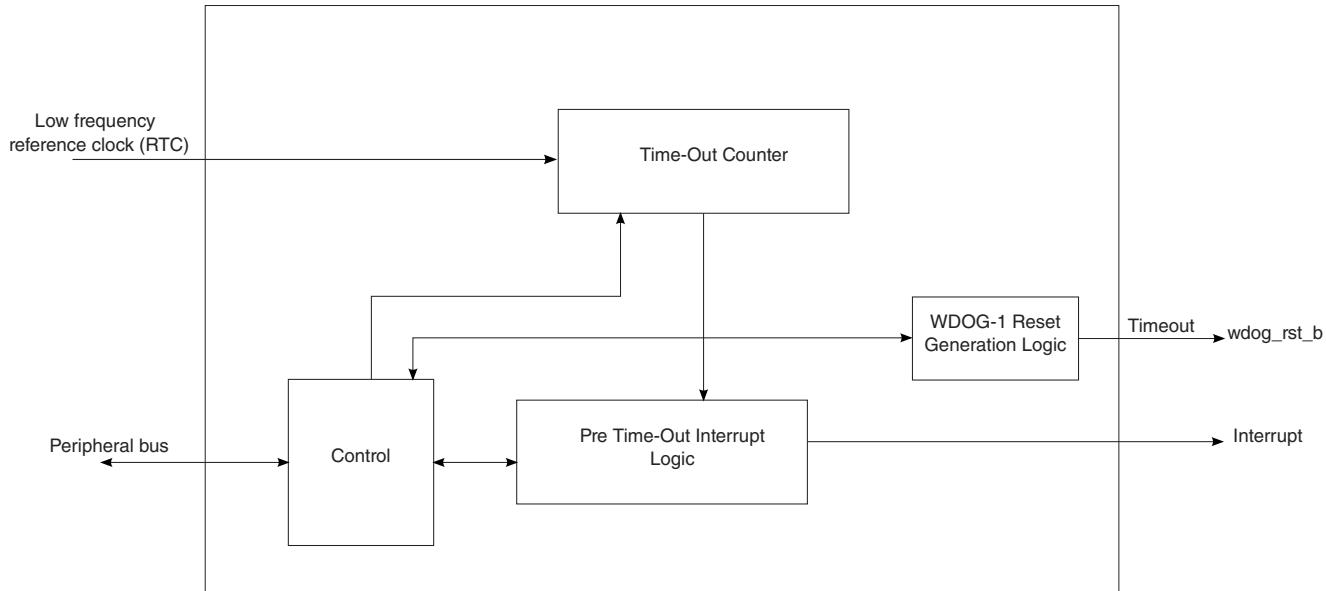
**NOTE**

There is no 1-to-1 mapping of WDOG counters with the cores. It is recommended that the secure-software should use WDOG2, while the non-secure software should use the rest of the blocks in the WDOG (that is core0 should use WDOG1, core1 should use WDOG3, core2 should use WDOG4, and core3 should use WDOG5). During reset, all the WDOG blocks are disabled.

The hardware will disable all the blocks in the WDOG only when all the four cores are disabled (in the PH20 state).

There is also a provision for WDOG signal assertion by time out counter expiration. There is an option of programmable interrupt generation before the counter actually times out. The time at which the interrupt needs to be generated prior to counter time out is programmable.

All WDOG modules use 32 KHz clock, driven at device input RTC pin for their counters. Flow diagrams for the timeout counter and interrupt operations are shown in [Figure 37-2](#)



**Figure 37-2. WDOG Diagram**

### 37.1.1 Features

The WDOG features are listed below:

- Configurable timeout counter with timeout periods from 0.5 to 128 seconds which, after timeout expiration, result in the assertion of timeout signal (wdog\_reset\_b).
- Time resolution of 0.5 seconds
- Programmable interrupt generation prior to timeout
- The duration between interrupt and timeout events can be programmed from 0 to 127.5 seconds in steps of 0.5 seconds.

## 37.2 Clocks

This section describes clocks and special clocking requirements of the block.

The WDOG uses the low frequency reference clock (RTC) for its counter and control operations.

The low frequency reference clock is a free-running clock and can't be gated.

## 37.3 Functional description

This section provides a complete functional description of the block.

### 37.3.1 Timeout event

The WDOG provides timeout periods from 0.5 to 128 seconds with a time resolution of 0.5 seconds.

The user can determine the timeout period by writing to the WDOG timeout field (WT[7:0]) in the [Watchdog Control Register \(WDOG\\_WCR\)](#). The WDOG must be enabled by setting the WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) for the timeout counter to start running. After the WDOG is enabled, the counter is activated, loads the timeout value and begins to count down from this programmed value. The timer will time out when the counter reaches zero and the WDOG outputs either a system reset signal, wdog\_rst\_b to COP.

However, the timeout condition can be prevented by reloading the counter with the new timeout value (WT[7:0] of WDOG\_WCR) if a service routine (see [Servicing WDOG to reload the counter](#)) is performed before the counter reaches zero. If any system errors occur which prevent the software from servicing the [Watchdog Service Register \(WDOG\\_WSR\)](#), the timeout condition occurs. By performing the service routine, the WDOG reloads its counter to the timeout value indicated by bits WT[7:0] of the [Watchdog Control Register \(WDOG\\_WCR\)](#) and it restarts the countdown.

A system reset will reset the counter and place it in the idle state at any time during the countdown. The counter flow diagram is shown in [Figure 37-4](#).

### **37.3.1.1 Servicing WDOG to reload the counter**

To reload a timeout value to the counter the proper service sequence begins by writing 0x\_5555 followed by 0x\_AAAA to the [Watchdog Service Register \(WDOG\\_WSR\)](#). Any number of instructions can be executed between the two writes. If the WDOG\_WSR is not loaded with 0x\_5555 prior to writing 0x\_AAAA to the WDOG\_WSR, the counter is not reloaded. If any value other than 0x\_AAAA is written to the WDOG\_WSR after 0x\_5555, the counter is not reloaded. This service sequence will reload the counter with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#). The timeout value can be changed at any point; it is reloaded when WDOG is serviced by the core.

### **37.3.2 Interrupt event**

Prior to timeout, the WDOG can generate an interrupt which can be considered a warning that timeout will occur shortly.

The duration between interrupt event and timeout event can be controlled by writing to the WICT field of [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#). It can vary between 0 and 127.5 seconds. If the WDOG is serviced ([Servicing WDOG to reload the counter](#)) before the interrupt generation, the counter will be reloaded with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#) and the interrupt will not be triggered.

### **37.3.3 Operations**

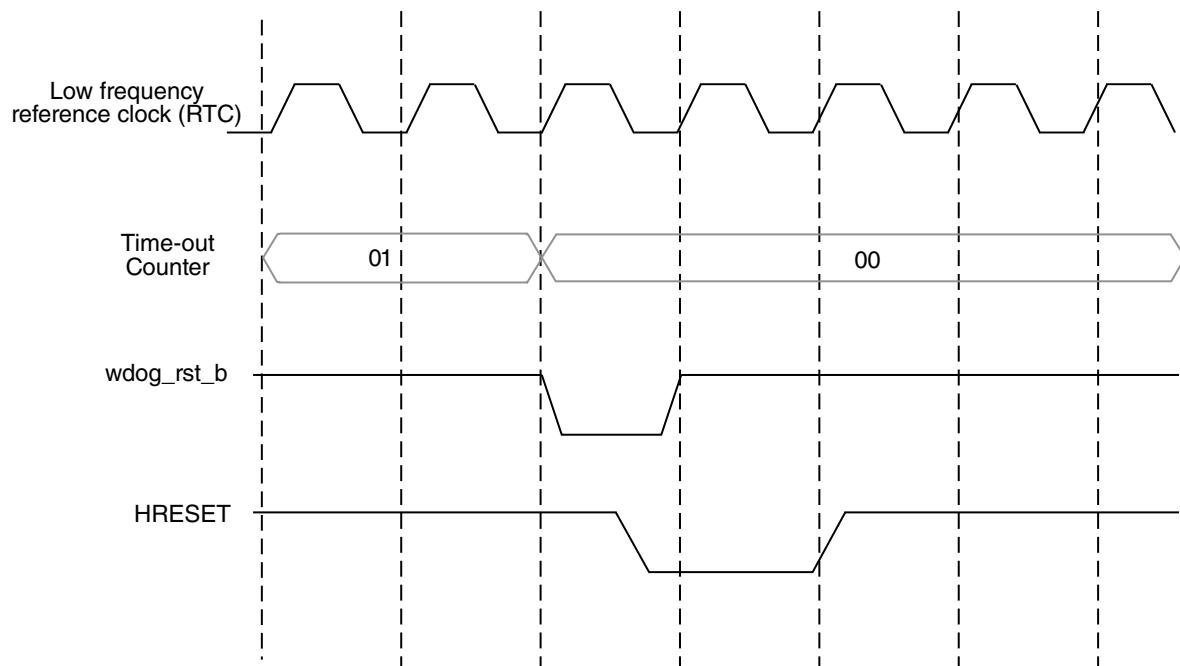
#### **37.3.3.1 Watchdog reset generation**

The WDOG generated reset signal wdog\_RST is asserted by the following operations:

- A software write to the Software Reset Signal (SRS) bit of the [Watchdog Control Register \(WDOG\\_WCR\)](#).
- WDOG timeout. See [Timeout event](#).

The wdog\_RST\_B will be asserted for one clock cycle of low frequency reference clock for both a timeout condition and a software write occurrence. It remains asserted for 1 clock cycle of low frequency reference clock even if a system reset is asserted in between.

The block is reset by a system reset and the WDOG counter will be disabled



**Figure 37-3. WDOG timeout/reset generation**

### 37.3.4 Interrupt

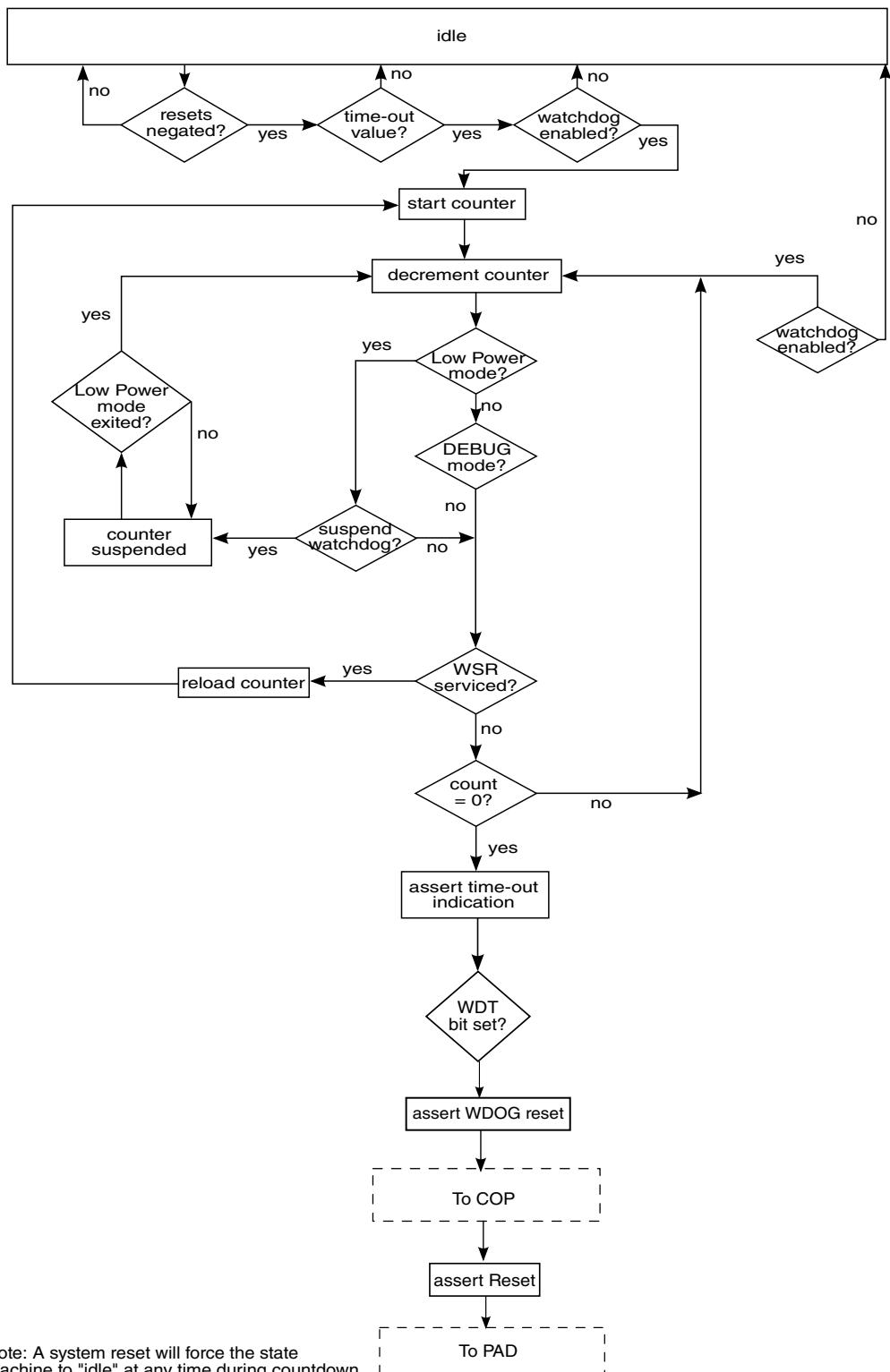
The WDOG has the feature of interrupt generation before timeout.

The interrupt will be generated only if the WIE bit in [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#) is set. The exact time at which the interrupt should occur (prior to timeout) depends on the value of WICT field of [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#). For example, if the WICT field has a value 0x04, then the interrupt will be generated two seconds prior to timeout. Once the interrupt is triggered the WTIS bit in [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#) will be set. The software needs to clear this bit to deassert the interrupt. If the WDOG is serviced before the interrupt generation then the counter will be reloaded with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#) and interrupt would not be triggered.

### 37.3.5 Flow Diagrams

A flow diagram of WDOG operation is shown below.

## Functional description



Note: A system reset will force the state machine to "idle" at any time during countdown.

**Figure 37-4. Time-Out Counter Flow Diagram**

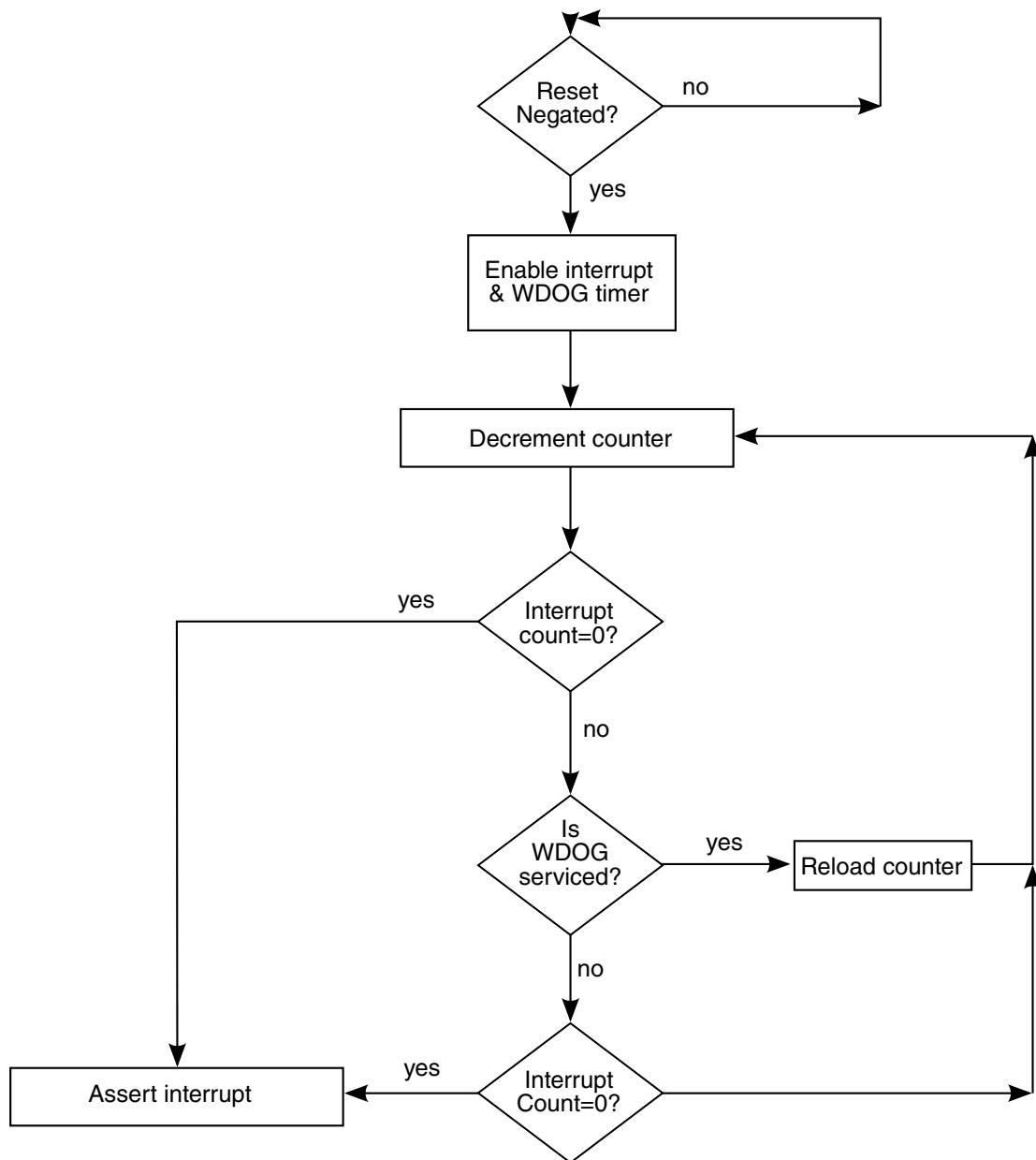


Figure 37-5. Interrupt Generation Flow Diagram

## 37.4 Initialization

The following sequence should be performed for WDOG initialization.

- WT field of [Watchdog Control Register \(WDOG\\_WCR\)](#) should be programmed for sufficient timeout value.
- WDOG should be enabled by setting WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) so that the timeout counter loads the WT field value of [Watchdog Control Register \(WDOG\\_WCR\)](#) and starts counting.

## 37.5 WDOG Memory Map/Register Definition

The WDOG has user-accessible, 16-bit registers used to configure, operate, and monitor the state of the watchdog timer. Byte operations can be performed on these registers. A 32-Bit access should be avoided, as the system may go to an unknown state.

**WDOG memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
2A7_0000	Watchdog Control Register (WDOG3_WCR)	16	R/W	0030h	<a href="#">37.5.1/2486</a>
2A7_0002	Watchdog Service Register (WDOG3_WSR)	16	R/W	0000h	<a href="#">37.5.2/2488</a>
2A7_0004	Watchdog Reset Status Register (WDOG3_WRSR)	16	R	0010h	<a href="#">37.5.3/2488</a>
2A7_0006	Watchdog Interrupt Control Register (WDOG3_WICR)	16	R/W	0004h	<a href="#">37.5.4/2489</a>
2A8_0000	Watchdog Control Register (WDOG4_WCR)	16	R/W	0030h	<a href="#">37.5.1/2486</a>
2A8_0002	Watchdog Service Register (WDOG4_WSR)	16	R/W	0000h	<a href="#">37.5.2/2488</a>
2A8_0004	Watchdog Reset Status Register (WDOG4_WRSR)	16	R	0010h	<a href="#">37.5.3/2488</a>
2A8_0006	Watchdog Interrupt Control Register (WDOG4_WICR)	16	R/W	0004h	<a href="#">37.5.4/2489</a>
2A9_0000	Watchdog Control Register (WDOG5_WCR)	16	R/W	0030h	<a href="#">37.5.1/2486</a>
2A9_0002	Watchdog Service Register (WDOG5_WSR)	16	R/W	0000h	<a href="#">37.5.2/2488</a>
2A9_0004	Watchdog Reset Status Register (WDOG5_WRSR)	16	R	0010h	<a href="#">37.5.3/2488</a>
2A9_0006	Watchdog Interrupt Control Register (WDOG5_WICR)	16	R/W	0004h	<a href="#">37.5.4/2489</a>
2AD_0000	Watchdog Control Register (WDOG1_WCR)	16	R/W	0030h	<a href="#">37.5.1/2486</a>
2AD_0002	Watchdog Service Register (WDOG1_WSR)	16	R/W	0000h	<a href="#">37.5.2/2488</a>
2AD_0004	Watchdog Reset Status Register (WDOG1_WRSR)	16	R	0010h	<a href="#">37.5.3/2488</a>
2AD_0006	Watchdog Interrupt Control Register (WDOG1_WICR)	16	R/W	0004h	<a href="#">37.5.4/2489</a>
2AE_0000	Watchdog Control Register (WDOG2_WCR)	16	R/W	0030h	<a href="#">37.5.1/2486</a>
2AE_0002	Watchdog Service Register (WDOG2_WSR)	16	R/W	0000h	<a href="#">37.5.2/2488</a>
2AE_0004	Watchdog Reset Status Register (WDOG2_WRSR)	16	R	0010h	<a href="#">37.5.3/2488</a>
2AE_0006	Watchdog Interrupt Control Register (WDOG2_WICR)	16	R/W	0004h	<a href="#">37.5.4/2489</a>

### 37.5.1 Watchdog Control Register (WDOGx\_WCR)

The Watchdog Control Register (WDOG\_WCR) controls the WDOG operation.

- WDZST is write-once only bit. Once the software does a write access to this bit, it will be locked and cannot be reprogrammed until the next system reset assertion.
- WDE is a write one once only bit. Once software performs a write "1" operation to this bit it cannot be reset/cleared until the next system reset (HRESET).

Address: Base address + 0h offset

Bit	0	1	2	3	4	5	6	7
Read	WT							
Write								
Reset	0	0	0	0	0	0	0	0
Bit	8	9	10	11	12	13	14	15
Read	0	Reserved			SRS	Reserved	WDE	Reserved
Write	0							WDZST
Reset	0	0	1	1	0	0	0	0

### WDOGx\_WCR field descriptions

Field	Description												
0–7 WT	<p>Watchdog Time-out Field. This 8-bit field contains the time-out value that is loaded into the Watchdog counter after the service routine has been performed or after the Watchdog is enabled. After reset, WT[7:0] must have a value written to it before enabling the Watchdog otherwise count value of zero which is 0.5 seconds is loaded into the counter.</p> <p><b>NOTE:</b> The time-out value can be written at any point of time but it is loaded to the counter at the time when WDOG is enabled or after the service routine has been performed. For more information see <a href="#">Timeout event</a>.</p> <table> <tr> <td>0x00</td> <td>0.5 Seconds (Default)</td> </tr> <tr> <td>0x01</td> <td>1.0 Seconds</td> </tr> <tr> <td>0x02</td> <td>1.5 Seconds</td> </tr> <tr> <td>0x03</td> <td>2.0 Seconds</td> </tr> <tr> <td>...</td> <td></td> </tr> <tr> <td>0xff</td> <td>128 Seconds</td> </tr> </table>	0x00	0.5 Seconds (Default)	0x01	1.0 Seconds	0x02	1.5 Seconds	0x03	2.0 Seconds	...		0xff	128 Seconds
0x00	0.5 Seconds (Default)												
0x01	1.0 Seconds												
0x02	1.5 Seconds												
0x03	2.0 Seconds												
...													
0xff	128 Seconds												
8 Reserved	This read-only field is reserved and always has the value 0.												
9–10 -	This field is reserved.												
11 SRS	<p>Software Reset Signal. Controls the software assertion of the WDOG-generated reset signal wdog_rst_b. This bit automatically resets to "1" after it has been asserted to "0".</p> <p><b>NOTE:</b> This bit does not generate the software reset to the block.</p> <table> <tr> <td>0</td> <td>Assert wdog_rst_b to COP</td> </tr> <tr> <td>1</td> <td>No effect on the system (Default)</td> </tr> </table>	0	Assert wdog_rst_b to COP	1	No effect on the system (Default)								
0	Assert wdog_rst_b to COP												
1	No effect on the system (Default)												
12 -	This field is reserved.												
13 WDE	<p>Watchdog Enable. Enables or disables the WDOG block. This is a write one once only bit. It is not possible to clear this bit by a software write, once the bit is set.</p> <p><b>NOTE:</b> This bit can be set/reset in debug mode (exception).</p> <table> <tr> <td>0</td> <td>Disable the Watchdog (Default)</td> </tr> <tr> <td>1</td> <td>Enable the Watchdog</td> </tr> </table>	0	Disable the Watchdog (Default)	1	Enable the Watchdog								
0	Disable the Watchdog (Default)												
1	Enable the Watchdog												
14 -	This field is reserved.												
15 WDZST	Watchdog Low Power. Determines the operation of the WDOG during low-power modes. This bit is write once-only.												

*Table continues on the next page...*

**WDOGx\_WCR field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> The WDOG can continue/suspend the timer operation in the low-power modes (STOP mode).</p> <p>0 Continue timer operation (Default) 1 Suspend the watchdog timer</p>

**37.5.2 Watchdog Service Register (WDOGx\_WSR)**

When enabled, the WDOG requires that a service sequence be written to the Watchdog Service Register (WSR) to prevent the timeout condition. The write access to this register is with one wait state, provided that the write data is 0xaaaa.

**NOTE**

Executing the service sequence will reload the WDOG timeout counter.

Address: Base address + 2h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read									WSR							
Write	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**WDOGx\_WSR field descriptions**

Field	Description
0–15 WSR	<p>Watchdog Service Register. This 16-bit field contains the Watchdog service sequence. Both writes must occur in the order listed prior to the time-out, but any number of instructions can be executed between the two writes. The service sequence must be performed as follows:</p> <p>0x5555 Write to the Watchdog Service Register (WDOG_WSR) 0xAAAA Write to the Watchdog Service Register (WDOG_WSR)</p>

**37.5.3 Watchdog Reset Status Register (WDOGx\_WRSR)**

The WRSR is a read-only register that records the source of the output reset assertion. It is not cleared by a hard reset. Therefore, only one bit in the WRSR will always be asserted high. The register will always indicate the source of the last reset generated due to WDOG. Read access to this register is with one wait state.

A reset can be generated by the following sources, as listed in priority from highest to lowest:

- Watchdog Time-out
- Software Reset

Address: Base address + 4h offset

Bit	0	1	2	3	4	5	6	7
Read					Reserved			
Write								
Reset	0	0	0	0	0	0	0	0
Bit	8	9	10	11	12	13	14	15
Read							TOUT	SFTW
Write				Reserved				
Reset	0	0	0	1	0	0	0	0

**WDOGx\_WRSR field descriptions**

Field	Description
0–13 -	This field is reserved.
14 TOUT	Timeout. Indicates whether the reset is the result of a WDOG timeout. 0 Reset is not the result of a WDOG timeout 1 Reset is the result of a WDOG timeout
15 SFTW	Software Reset. Indicates whether the reset is the result of a WDOG software reset by asserting SRS bit 0 Reset is not the result of a software reset 1 Reset is the result of a software reset

**37.5.4 Watchdog Interrupt Control Register (WDOGx\_WICR)**

The WDOG\_WICR controls the WDOG interrupt generation.

Address: Base address + 6h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read		WTIS			0											
Write	WIE	w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

**WDOGx\_WICR field descriptions**

Field	Description
0 WIE	Watchdog timer interrupt enable bit. Reset value is 0. <b>NOTE:</b> This bit is a write once only bit. Once the software does a write access to this bit, it will get locked and cannot be reprogrammed until the next system reset assertion 0 Disable Interrupt (Default) 1 Enable Interrupt

*Table continues on the next page...*

**WDOGx\_WICR field descriptions (continued)**

Field	Description
1 WTIS	Watchdog Timer Interrupt Status bit will reflect the timer interrupt status, whether interrupt has occurred or not. Once the interrupt has been triggered software must clear this bit by writing 1 to it.  0 No interrupt has occurred (Default) 1 Interrupt has occurred
2–7 Reserved	This read-only field is reserved and always has the value 0.
8–15 WICT	Watchdog Interrupt Count Time-out (WICT) field determines, how long before the counter time-out must the interrupt occur. The reset value is 0x04 implies interrupt will occur 2 seconds before time-out. The maximum value that can be programmed to WICT field is 127.5 seconds with a resolution of 0.5 seconds.  <b>NOTE:</b> This field is write once only. Once the software does a write access to this field, it will get locked and cannot be reprogrammed until the next system reset assertion.  0x00 Time duration between interrupt and time-out is 0 seconds 0x01 Time duration between interrupt and time-out is 0.5 seconds 0x04 Time duration between interrupt and time-out is 2 seconds (Default) ... 0xff Time duration between interrupt and time-out is 127.5 seconds

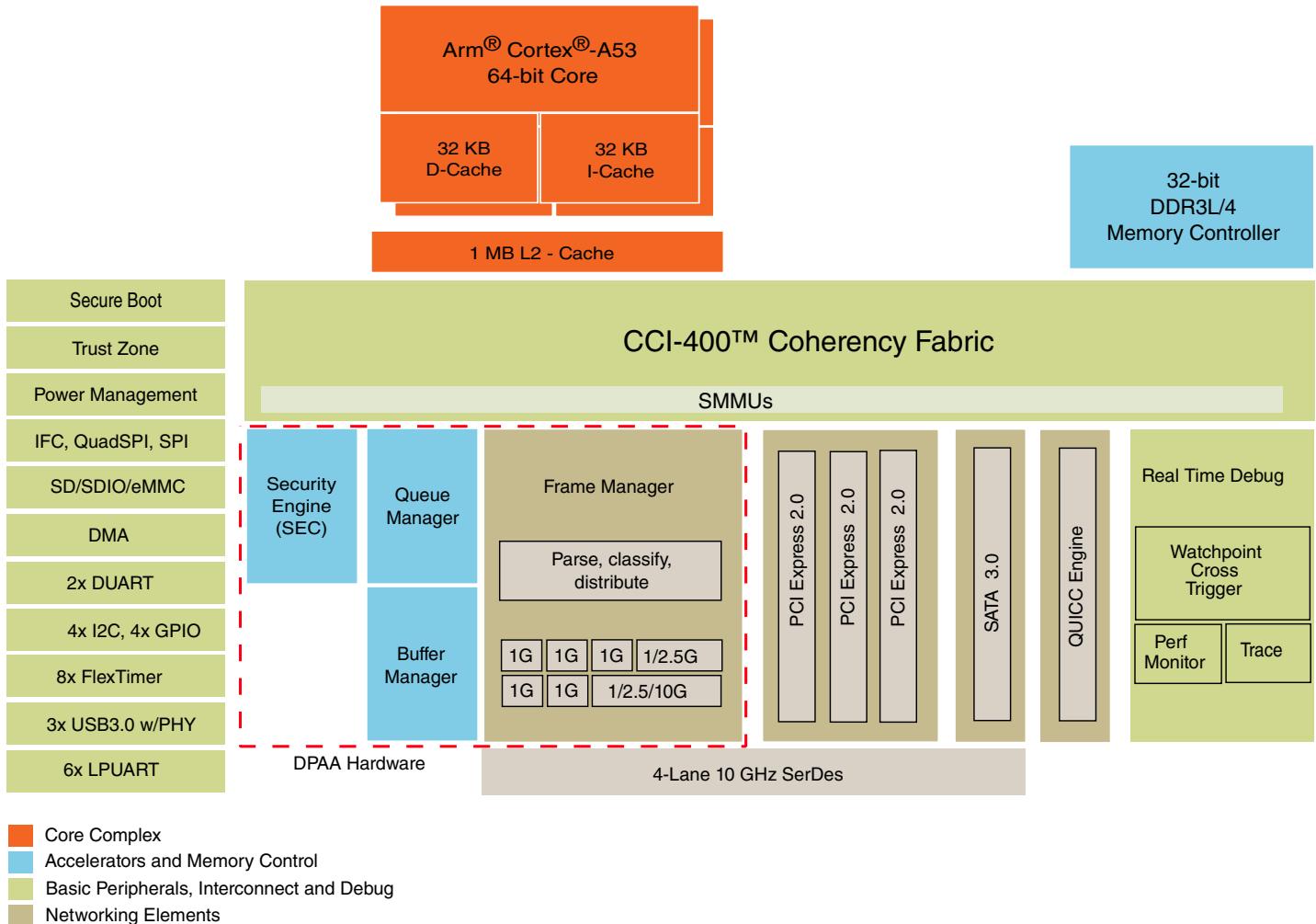
# **Appendix A**

## **LS1023A - A Dual Core Version**

### **A.1 Overview**

The LS1043A is also available in dual core version known as LS1023A.

This figure shows the major functional units within the chip.



**Figure A-1. LS1023A block diagram**

# Appendix B

## Appendix LS1043A (23x23 package)

### B.1 Introduction

The appendix lists the changes applicable to LS1043A 23x23 package. The specifications provided here supersede to those available for LS1043A 21x21 package.

### B.2 Signals Overview

This following table provides the pinout listing for the LS1043A (23x23) by bus. This table details the signal name, interface, alternate functions, number of signals, and whether the signal is an input, output, or bidirectional. The direction of the multiplexed signals applies for the primary signal function listed in the left-most column of the table for that row (and does not apply for the state of the reset configuration signals). Finally, the tables provide a pointer to the table where the signal function is described.

The primary signals are **bold** in the table.

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block**

Name	Description	Alternate Function(s)	Pin type
<b>DDR SDRAM Memory Interface 1</b>			
D1_MA00	Address	-	O
D1_MA01	Address	-	O
D1_MA02	Address	-	O
D1_MA03	Address	-	O
D1_MA04	Address	-	O
D1_MA05	Address	-	O
D1_MA06	Address	-	O
D1_MA07	Address	-	O
D1_MA08	Address	-	O

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
D1_MA09	Address	-	O
D1_MA10	Address	-	O
D1_MA11	Address	-	O
D1_MA12	Address	-	O
D1_MA13	Address	-	O
D1_MACT_B	Address	-	O
D1_MALERT_B	Address Parity Error		I
D1_MBA0	Bank Select	-	O
D1_MBA1	Bank Select	-	O
D1_MB戈0	Bank Select	-	O
D1_MB戈1	Address	-	O
D1_MCAS_B	Column Address Strobe	-	O
D1_MCK0	Clock	-	O
D1_MCK0_B	Clock Complement	-	O
D1_MCK1	Clock	-	O
D1_MCK1_B	Clock Complement	-	O
D1_MCKE0	Clock Enable	-	O
D1_MCKE1	Clock Enable	-	O
D1_MCS0_B	Chip Select	-	O
D1_MCS1_B	Chip Select	-	O
D1_MCS2_B	Chip Select	-	O
D1_MCS3_B	Chip Select	-	O
D1_MDIC0	Driver Impedance Calibration	-	IO
D1_MDIC1	Driver Impedance Calibration	-	IO
D1_MDM0	Data Mask	-	O
D1_MDM1	Data Mask	-	O
D1_MDM2	Data Mask	-	O
D1_MDM3	Data Mask	-	O
D1_MDM8	Data Mask	-	O
D1_MDQ00	Data	-	IO
D1_MDQ01	Data	-	IO
D1_MDQ02	Data	-	IO
D1_MDQ03	Data	-	IO
D1_MDQ04	Data	-	IO
D1_MDQ05	Data	-	IO
D1_MDQ06	Data	-	IO
D1_MDQ07	Data	-	IO
D1_MDQ08	Data	-	IO
D1_MDQ09	Data	-	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
D1_MDQ10	Data	-	IO
D1_MDQ11	Data	-	IO
D1_MDQ12	Data	-	IO
D1_MDQ13	Data	-	IO
D1_MDQ14	Data	-	IO
D1_MDQ15	Data	-	IO
D1_MDQ16	Data	-	IO
D1_MDQ17	Data	-	IO
D1_MDQ18	Data	-	IO
D1_MDQ19	Data	-	IO
D1_MDQ20	Data	-	IO
D1_MDQ21	Data	-	IO
D1_MDQ22	Data	-	IO
D1_MDQ23	Data	-	IO
D1_MDQ24	Data	-	IO
D1_MDQ25	Data	-	IO
D1_MDQ26	Data	-	IO
D1_MDQ27	Data	-	IO
D1_MDQ28	Data	-	IO
D1_MDQ29	Data	-	IO
D1_MDQ30	Data	-	IO
D1_MDQ31	Data	-	IO
D1_MDQS0	Data Strobe	-	IO
D1_MDQS0_B	Data Strobe	-	IO
D1_MDQS1	Data Strobe	-	IO
D1_MDQS1_B	Data Strobe	-	IO
D1_MDQS2	Data Strobe	-	IO
D1_MDQS2_B	Data Strobe	-	IO
D1_MDQS3	Data Strobe	-	IO
D1_MDQS3_B	Data Strobe	-	IO
D1_MDQS8	Data Strobe	-	IO
D1_MDQS8_B	Data Strobe	-	IO
D1_MECC0	Error Correcting Code	-	IO
D1_MECC1	Error Correcting Code	-	IO
D1_MECC2	Error Correcting Code	-	IO
D1_MECC3	Error Correcting Code	-	IO
D1_MODT0	On Die Termination	-	O
D1_MODT1	On Die Termination	-	O
D1_MPAR	Address Parity Out	-	O

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
D1_MRAS_B	Row Address Strobe	-	O
D1_MWE_B	Write Enable	-	O
<b>Integrated Flash Controller</b>			
IFC_A16	IFC Address	QSPI_A_CS0	O
IFC_A17	IFC Address	QSPI_A_CS1	O
IFC_A18	IFC Address	QSPI_A_SCK	O
IFC_A19	IFC Address	QSPI_B_CS0	O
IFC_A20	IFC Address	QSPI_B_CS1	O
IFC_A21	IFC Address	QSPI_B_SCK cfg_dram_type	O
IFC_A22 / IFC_WP1_B	IFC Address	QSPI_A_DATA0	O
IFC_A23 / IFC_WP2_B	IFC Address	QSPI_A_DATA1	O
IFC_A24 / IFC_WP3_B	IFC Address	QSPI_A_DATA2	O
IFC_A25 / IFC_CS4_B / IFC_RB2_B	IFC Address	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	O
IFC_A26 / IFC_CS5_B / IFC_RB3_B	IFC Address	GPIO2_26 FTM5_CH1	O
IFC_A27 / IFC_CS6_B	IFC Address	GPIO2_27 FTM5_EXTCLK	O
IFC_AD00	IFC Address / Data	cfg_gpinput0	IO
IFC_AD01	IFC Address / Data	cfg_gpinput1	IO
IFC_AD02	IFC Address / Data	cfg_gpinput2	IO
IFC_AD03	IFC Address / Data	cfg_gpinput3	IO
IFC_AD04	IFC Address / Data	cfg_gpinput4	IO
IFC_AD05	IFC Address / Data	cfg_gpinput5	IO
IFC_AD06	IFC Address / Data	cfg_gpinput6	IO
IFC_AD07	IFC Address / Data	cfg_gpinput7	IO
IFC_AD08	IFC Address / Data	cfg_rcw_src0	IO
IFC_AD09	IFC Address / Data	cfg_rcw_src1	IO
IFC_AD10	IFC Address / Data	cfg_rcw_src2	IO
IFC_AD11	IFC Address / Data	cfg_rcw_src3	IO
IFC_AD12	IFC Address / Data	cfg_rcw_src4	IO
IFC_AD13	IFC Address / Data	cfg_rcw_src5	IO
IFC_AD14	IFC Address / Data	cfg_rcw_src6	IO
IFC_AD15	IFC Address / Data	cfg_rcw_src7	IO
IFC_AVD	IFC Address Valid		O
IFC_BCTL	IFC Buffer control		O
IFC_CLE	IFC Command Latch Enable / Write Enable	cfg_rcw_src8	O
IFC_CLK0	IFC Clock	-	O

Table continues on the next page...

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>IFC_CLK1</b>	IFC Clock	-	O
<b>IFC_CS0_B</b>	IFC Chip Select	-	O
<b>IFC_CS1_B</b>	IFC Chip Select	GPIO2_10 FTM7_CH0	O
<b>IFC_CS2_B</b>	IFC Chip Select	GPIO2_11 FTM7_CH1	O
<b>IFC_CS3_B</b>	IFC Chip Select	GPIO2_12 QSPI_B_DATA3 FTM7_EXTCLK	O
<b>IFC_CS4_B / IFC_A25 / IFC_RB2_B</b>	IFC Chip Select	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	O
<b>IFC_CS5_B / IFC_A26 / IFC_RB3_B</b>	IFC Chip Select	GPIO2_26 FTM5_CH1	O
<b>IFC_CS6_B / IFC_A27</b>	IFC Chip Select	GPIO2_27 FTM5_EXTCLK	O
<b>IFC_NDDDR_CLK</b>	IFC NAND DDR Clock	-	O
<b>IFC_NDDQS</b>	IFC DQS Strobe	-	IO
<b>IFC_OE_B</b>	IFC Output Enable	cfg_eng_use1	O
<b>IFC_PAR0</b>	IFC Address & Data Parity	GPIO2_13 QSPI_B_DATA0 FTM6_CH0	IO
<b>IFC_PAR1</b>	IFC Address & Data Parity	GPIO2_14 QSPI_B_DATA1 FTM6_CH1	IO
<b>IFC_PERR_B</b>	IFC Parity Error	GPIO2_15 QSPI_B_DATA2 FTM6_EXTCLK	I
<b>IFC_RB0_B</b>	IFC Ready / Busy CS0		I
<b>IFC_RB1_B</b>	IFC Ready / Busy CS1		I
<b>IFC_RB2_B / IFC_A25 / IFC_CS4_B</b>	IFC Ready/Busy CS 2	GPIO2_25 QSPI_A_DATA3 FTM5_CH0	I
<b>IFC_RB3_B / IFC_A26 / IFC_CS5_B</b>	IFC Ready/Busy CS 3	GPIO2_26 FTM5_CH1	I
<b>IFC_TE</b>	IFC External Transceiver Enable	cfg_ifc_te	O
<b>IFC_WE0_B</b>	IFC Write Enable	cfg_eng_use0	O
<b>IFC_WP0_B</b>	IFC Write Protect	cfg_eng_use2	O
<b>IFC_WP1_B / IFC_A22</b>	IFC Write Protect	QSPI_A_DATA0	O
<b>IFC_WP2_B / IFC_A23</b>	IFC Write Protect	QSPI_A_DATA1	O
<b>IFC_WP3_B / IFC_A24</b>	IFC Write Protect	QSPI_A_DATA2	O
<b>DUART</b>			
<b>UART1_CTS_B / UART3_SIN</b>	Clear To Send	GPIO1_21 FTM4_CH4	I

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		LPUART2_SIN	
<b>UART1_RTS_B / UART3_SOUT</b>	Ready to Send	GPIO1_19 LPUART2_SOUT FTM4_CH2	O
<b>UART1_SIN</b>	Receive Data	GPIO1_17	I
<b>UART1_SOUT</b>	Transmit Data	GPIO1_15	O
<b>UART2_CTS_B / UART4_SIN</b>	Clear To Send	GPIO1_22 FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	I
<b>UART2_RTS_B / UART4_SOUT</b>	Ready to Send	GPIO1_20 LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	O
<b>UART2_SIN</b>	Receive Data	GPIO1_18 FTM4_CH1 LPUART1_SIN	I
<b>UART2_SOUT</b>	Transmit Data	GPIO1_16 LPUART1_SOUT FTM4_CH0	O
<b>UART3_SIN / UART1_CTS_B</b>	Receive Data	GPIO1_21 FTM4_CH4 LPUART2_SIN	I
<b>UART3_SOUT / UART1_RTS_B</b>	Transmit Data	GPIO1_19 LPUART2_SOUT FTM4_CH2	O
<b>UART4_SIN / UART2_CTS_B</b>	Receive Data	GPIO1_22 FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	I
<b>UART4_SOUT / UART2_RTS_B</b>	Transmit Data	GPIO1_20 LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	O
<b>SPI Interface</b>			
<b>SPI_PCS0</b>	SPI Chip Select	GPIO2_00 SDHC_DAT4 SDHC_VS	O
<b>SPI_PCS1</b>	SPI Chip Select	GPIO2_01 SDHC_DAT5 SDHC_CMD_DIR	O
<b>SPI_PCS2</b>	SPI Chip Select	GPIO2_02 SDHC_DAT6 SDHC_DAT0_DIR	O
<b>SPI_PCS3</b>	SPI Chip Select	GPIO2_03 SDHC_DAT7 SDHC_DAT123_DIR	O

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>SPI_SCK</b>	SPI Clock		O
<b>SPI_SIN</b>	Master In Slave Out	SDHC_CLK_SYNC_IN	I
<b>SPI_SOUT</b>	Master Out Slave In	SDHC_CLK_SYNC_OUT	IO
<b>eSDHC</b>			
<b>SDHC_CD_B</b>	Command	IIC2_SCL GPIO4_02 FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	I
<b>SDHC_CLK</b>	Host to Card Clock	GPIO2_09 LPUART3_CTS_B LPUART6_SIN FTM4_QD_PHB	O
<b>SDHC_CLK_SYNC_IN</b>	IN	<b>SPI_SIN</b>	I
<b>SDHC_CLK_SYNC_OUT</b>	OUT	<b>SPI_SOUT</b>	O
<b>SDHC_CMD</b>	Command/Response	GPIO2_04 LPUART3_SOUT FTM4_CH6	IO
<b>SDHC_CMD_DIR</b>	DIR	<b>SPI_PCS1</b> GPIO2_01 SDHC_DAT5	O
<b>SDHC_DAT0</b>	Data	GPIO2_05 FTM4_CH7 LPUART3_SIN	IO
<b>SDHC_DAT0_DIR</b>	DIR	<b>SPI_PCS2</b> GPIO2_02 SDHC_DAT6	O
<b>SDHC_DAT1</b>	Data	GPIO2_06 LPUART5_SOUT FTM4_FAULT LPUART2_RTS_B	IO
<b>SDHC_DAT123_DIR</b>	DIR	<b>SPI_PCS3</b> GPIO2_03 SDHC_DAT7	O
<b>SDHC_DAT2</b>	Data	GPIO2_07 LPUART2_CTS_B LPUART5_SIN FTM4_EXTCLK	IO
<b>SDHC_DAT3</b>	Data	GPIO2_08 LPUART6_SOUT FTM4_QD_PHA LPUART3_RTS_B	IO
<b>SDHC_DAT4</b>	Data	<b>SPI_PCS0</b> GPIO2_00 SDHC_VS	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
SDHC_DAT5	Data	SPI_PCS1 GPIO2_01 SDHC_CMD_DIR	IO
SDHC_DAT6	Data	SPI_PCS2 GPIO2_02 SDHC_DAT0_DIR	IO
SDHC_DAT7	Data	SPI_PCS3 GPIO2_03 SDHC_DAT123_DIR	IO
SDHC_VS	VS	SPI_PCS0 GPIO2_00 SDHC_DAT4	O
SDHC_WP	Write Protect	IIC2_SDA GPIO4_03 FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	I
<b>Programmable Interrupt Controller</b>			
EVT9_B	Interrupt Output		O
IRQ00	External Interrupt		I
IRQ01	External Interrupt		I
IRQ02	External Interrupt		I
IRQ03	External Interrupt	GPIO1_23 FTM3_CH7 TDMA_TSYNC UC3_RTSB_TXEN	I
IRQ04	External Interrupt	GPIO1_24 FTM3_CH0 TDMA_RXD UC1_RXD7 TDMA_TXD	I
IRQ05	External Interrupt	GPIO1_25 FTM3_CH1 TDMA_RSYNC UC1_CTSB_RXDV	I
IRQ06	External Interrupt	GPIO1_26 FTM3_CH2 TDMA_RXD_EXC TDMA_TXD UC1_TXD7	I
IRQ07	External Interrupt	GPIO1_27 FTM3_CH3 TDMA_TSYNC UC1_RTSB_TXEN	I
IRQ08	External Interrupt	GPIO1_28 FTM3_CH4	I

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMB_RXD UC3_RXD7 TDMB_TXD	
<b>IRQ09</b>	External Interrupt	GPIO1_29 FTM3_CH5 TDMB_RSYNC UC3_CTSB_RXDV	I
<b>IRQ10</b>	External Interrupt	GPIO1_30 FTM3_CH6 TDMB_RXD_EXC TDMB_TXD UC3_TXD7	I
<b>IRQ11</b>	External Interrupt	GPIO1_31	I
<b>Battery Backed Trust</b>			
<b>TA_BB_TMP_DETECT_B</b>	Battery Backed Tamper Detect	-	I
<b>Trust</b>			
<b>TA_TMP_DETECT_B</b>	Tamper Detect		I
<b>System Control</b>			
<b>HRESET_B</b>	Hard Reset	-	IO
<b>PORESET_B</b>	Power On Reset	-	I
<b>RESET_REQ_B</b>	Reset Request (POR or Hard)		O
<b>Power Management</b>			
<b>ASLEEP</b>	Asleep	GPIO1_13	O
<b>SYSCLK</b>			
<b>SYSCLK</b>	System Clock	-	I
<b>DDR Clocking</b>			
<b>DDRCLK</b>	DDR Controller Clock	-	I
<b>RTC</b>			
<b>RTC</b>	Real Time Clock	GPIO1_14	I
<b>Debug</b>			
<b>CKSTP_OUT_B</b>	Reserved	-	O
<b>CLK_OUT</b>	Clock Out	-	O
<b>EVT0_B</b>	Event 0	-	IO
<b>EVT1_B</b>	Event 1	-	IO
<b>EVT2_B</b>	Event 2	-	IO
<b>EVT3_B</b>	Event 3	-	IO
<b>EVT4_B</b>	Event 4	-	IO
<b>EVT5_B</b>	Event 5	IIC3_SCL GPIO4_10 USB2_DRVVBUS BRGO4 FTM8_CH0	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		CLK11	
EVT6_B	Event 6	IIC3_SDA GPIO4_11 USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
EVT7_B	Event 7	IIC4_SCL GPIO4_12 USB3_DRVBUS TDMA_RQ FTM3_FAULT UC1_CDB_RXER	IO
EVT8_B	Event 8	IIC4_SDA GPIO4_13 USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
<b>DFT</b>			
JTAG_BSR_VSEL	An IEEE 1149.1 JTAG compliance enable pin. 0: Normal operation. 1: To be compliant to the 1149.1 specification for boundary scan functions. The JTAG compliant state is documented in the BSDL.  <b>NOTE:</b> When this pin is tied high, the USB PHY will not come out or reset in normal functional mode.	-	I
SCAN_MODE_B	Reserved	-	I
TBSCAN_EN_B	An IEEE 1149.1 JTAG compliance enable pin. 0:To be compliant to the 1149.1 specification for boundary scan functions. The JTAG compliant state is documented in the BSDL. 1: JTAG connects to DAP controller for the Arm core debug.	-	I
TEST_SEL_B	Reserved	-	I
<b>JTAG</b>			
TCK	Test Clock	-	I
TDI	Test Data In	-	I
TDO	Test Data Out	-	O
TMS	Test Mode Select	-	I
TRST_B	Test Reset	-	I
<b>Analog Signals</b>			
D1_MVREF	SSTL Reference Voltage	-	IO
D1_TPA	DDR Controller 1 Test Point Analog	-	IO
FA_ANALOG_G_V	Reserved	-	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>FA_ANALOG_PIN</b>	Reserved	-	IO
<b>TD1_ANODE</b>	Thermal diode anode	-	IO
<b>TD1_CATHODE</b>	Thermal diode cathode	-	IO
<b>TH_TPA</b>	Thermal Test Point Analog	-	-
<b>SerDes</b>			
<b>SD1_IMP_CAL_RX</b>	SerDes Receive Impedance Calibration	-	I
<b>SD1_IMP_CAL_TX</b>	SerDes Transmit Impedance Calibration	-	I
<b>SD1_PLL1_TPA</b>	SerDes PLL 1 Test Point Analog	-	O
<b>SD1_PLL1_TPD</b>	SerDes Test Point Digital	-	O
<b>SD1_PLL2_TPA</b>	SerDes PLL 2 Test Point Analog	-	O
<b>SD1_PLL2_TPD</b>	SerDes Test Point Digital	-	O
<b>SD1_REF_CLK1_N</b>	SerDes PLL 1 Reference Clock Complement	-	I
<b>SD1_REF_CLK1_P</b>	SerDes PLL 1 Reference Clock	-	I
<b>SD1_REF_CLK2_N</b>	SerDes PLL 2 Reference Clock Complement	-	I
<b>SD1_REF_CLK2_P</b>	SerDes PLL 2 Reference Clock	-	I
<b>SD1_RX0_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX0_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX1_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX1_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX2_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX2_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_RX3_N</b>	SerDes Receive Data (negative)	-	I
<b>SD1_RX3_P</b>	SerDes Receive Data (positive)	-	I
<b>SD1_TX0_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX0_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX1_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX1_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX2_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX2_P</b>	SerDes Transmit Data (positive)	-	O
<b>SD1_TX3_N</b>	SerDes Transmit Data (negative)	-	O
<b>SD1_TX3_P</b>	SerDes Transmit Data (positive)	-	O
<b>USB3 PHY 1</b>			
<b>USB1_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB1_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB1_ID</b>	USB PHY ID Detect	-	I
<b>USB1_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB1_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB1_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB1_TX_M</b>	USB PHY SS Transmit Data (-)	-	O

*Table continues on the next page...*

## Signals Overview

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>USB1_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB1_VBUS</b>	USB PHY VBUS	-	I
<b>USB3 PHY 2</b>			
<b>USB2_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB2_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB2_ID</b>	USB PHY ID Detect	-	I
<b>USB2_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB2_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB2_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB2_TX_M</b>	USB PHY SS Transmit Data (-)	-	O
<b>USB2_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB2_VBUS</b>	USB PHY VBUS	-	I
<b>USB3 PHY 3</b>			
<b>USB3_D_M</b>	USB PHY HS Data (-)	-	IO
<b>USB3_D_P</b>	USB PHY HS Data (+)	-	IO
<b>USB3_ID</b>	USB PHY ID Detect	-	I
<b>USB3_RESREF</b>	USB PHY Impedance Calibration	-	IO
<b>USB3_RX_M</b>	USB PHY SS Receive Data (-)	-	I
<b>USB3_RX_P</b>	USB PHY SS Receive Data (+)	-	I
<b>USB3_TX_M</b>	USB PHY SS Transmit Data (-)	-	O
<b>USB3_TX_P</b>	USB PHY SS Transmit Data (+)	-	O
<b>USB3_VBUS</b>	USB PHY VBUS	-	I
<b>Ethernet Management Interface 1</b>			
<b>EMI1_MDC</b>	Management Data Clock	GPIO3_00	O
<b>EMI1_MDIO</b>	Management Data In/Out	GPIO3_01	IO
<b>Ethernet Management Interface 2</b>			
<b>EMI2_MDC</b>	Management Data Clock	GPIO4_00	O
<b>EMI2_MDIO</b>	Management Data In/Out	GPIO4_01	IO
<b>Ethernet Controller 1</b>			
<b>EC1_GTX_CLK</b>	Transmit Clock Out	GPIO3_07 FTM1_EXTCLK	O
<b>EC1_GTX_CLK125</b>	Reference Clock	GPIO3_08	I
<b>EC1_RXD0</b>	Receive Data	GPIO3_12 FTM1_CH0	I
<b>EC1_RXD1</b>	Receive Data	GPIO3_11 FTM1_CH1	I
<b>EC1_RXD2</b>	Receive Data	GPIO3_10 FTM1_CH6	I
<b>EC1_RXD3</b>	Receive Data	GPIO3_09 FTM1_CH4	I

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>EC1_RX_CLK</b>	Receive Clock	GPIO3_13 FTM1_QD_PHA	I
<b>EC1_RX_DV</b>	Receive Data Valid	GPIO3_14 FTM1_QD_PHB	I
<b>EC1_TXD0</b>	Transmit Data	GPIO3_05 FTM1_CH2	O
<b>EC1_TXD1</b>	Transmit Data	GPIO3_04 FTM1_CH3	O
<b>EC1_TXD2</b>	Transmit Data	GPIO3_03 FTM1_CH7	O
<b>EC1_TXD3</b>	Transmit Data	GPIO3_02 FTM1_CH5	O
<b>EC1_TX_EN</b>	Transmit Enable	GPIO3_06 FTM1_FAULT	O
<b>Ethernet Controller 2</b>			
<b>EC2_GTX_CLK</b>	Transmit Clock Out	GPIO3_20 FTM2_EXTCLK	O
<b>EC2_GTX_CLK125</b>	Reference Clock	GPIO3_21	I
<b>EC2_RXD0</b>	Receive Data	GPIO3_25 TSEC_1588_TRIG _IN2 FTM2_CH0	I
<b>EC2_RXD1</b>	Receive Data	GPIO3_24 TSEC_1588_PULS E_OUT1 FTM2_CH1	I
<b>EC2_RXD2</b>	Receive Data	GPIO3_23 FTM2_CH6	I
<b>EC2_RXD3</b>	Receive Data	GPIO3_22 FTM2_CH4	I
<b>EC2_RX_CLK</b>	Receive Clock	GPIO3_26 TSEC_1588_CLK_I N FTM2_QD_PHA	I
<b>EC2_RX_DV</b>	Receive Data Valid	GPIO3_27 TSEC_1588_TRIG _IN1 FTM2_QD_PHB	I
<b>EC2_TXD0</b>	Transmit Data	GPIO3_18 TSEC_1588_PULS E_OUT2 FTM2_CH2	O
<b>EC2_TXD1</b>	Transmit Data	GPIO3_17 TSEC_1588_CLK_ OUT FTM2_CH3	O
<b>EC2_TXD2</b>	Transmit Data	GPIO3_16	O

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TSEC_1588_ALAR M_OUT1 FTM2_CH7	
<b>EC2_TXD3</b>	Transmit Data	GPIO3_15 TSEC_1588_ALAR M_OUT2 FTM2_CH5	O
<b>EC2_TX_EN</b>	Transmit Enable	GPIO3_19 FTM2_FAULT	O
<b>I2C</b>			
<b>IIC1_SCL</b>	Serial Clock (supports PBL)		IO
<b>IIC1_SDA</b>	Serial Data (supports PBL)		IO
<b>IIC2_SCL</b>	Serial Clock	GPIO4_02 SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	IO
<b>IIC2_SDA</b>	Serial Data	GPIO4_03 SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	IO
<b>IIC3_SCL</b>	Serial Clock	GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0 CLK11	IO
<b>IIC3_SDA</b>	Serial Data	GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
<b>IIC4_SCL</b>	Serial Clock	GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ FTM3_FAULT UC1_CDB_RXER	IO
<b>IIC4_SDA</b>	Serial Data	GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
<b>USB</b>			

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
USB2_DRVVBUS	DRV VBus	IIC3_SCL GPIO4_10 EVT5_B BRGO4 FTM8_CH0 CLK11	O
USB2_PWRFAULT	PWR Fault	IIC3_SDA GPIO4_11 EVT6_B BRGO1 FTM8_CH1 CLK12_CLK8	I
USB3_DRVVBUS	DRV Bus	IIC4_SCL GPIO4_12 EVT7_B TDMA_RQ FTM3_FAULT UC1_CDB_RXER	O
USB3_PWRFAULT	PWR Fault	IIC4_SDA GPIO4_13 EVT8_B TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	I
<b>USB_DRVVBUS</b>	<b>USB_DRVVBUS</b>	<b>GPIO4_29</b>	<b>O</b>
<b>USB_PWRFAULT</b>	<b>USB_PWRFAULT</b>	<b>GPIO4_30</b>	<b>I</b>
<b>Battery Backed RTC</b>			
TA_BB_RTC	Reserved	-	I
<b>DSYSCLK</b>			
DIFF_SYSCLK	Single Source System Clock Differential (positive)	-	I
DIFF_SYSCLK_B	Single Source System Clock Differential (negative)	-	I
<b>Power-On-Reset Configuration</b>			
cfg_dram_type	Power-on-Reset Configuration	IFC_A21 QSPI_B_SCK	I
cfg_eng_use0	Power-on-Reset Configuration	IFC_WE0_B	I
cfg_eng_use1	Power-on-Reset Configuration	IFC_OE_B	I
cfg_eng_use2	Power-on-Reset Configuration	IFC_WP0_B	I
cfg_gpininput0	Power-on-Reset Configuration	IFC_AD00	I
cfg_gpininput1	Power-on-Reset Configuration	IFC_AD01	I
cfg_gpininput2	Power-on-Reset Configuration	IFC_AD02	I
cfg_gpininput3	Power-on-Reset Configuration	IFC_AD03	I
cfg_gpininput4	Power-on-Reset Configuration	IFC_AD04	I
cfg_gpininput5	Power-on-Reset Configuration	IFC_AD05	I
cfg_gpininput6	Power-on-Reset Configuration	IFC_AD06	I
cfg_gpininput7	Power-on-Reset Configuration	IFC_AD07	I

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
cfg_ifc_te	Power-on-Reset Configuration	<b>IFC_TE</b>	I
cfg_rcw_src0	Power-on-Reset Configuration	<b>IFC_AD08</b>	I
cfg_rcw_src1	Power-on-Reset Configuration	<b>IFC_AD09</b>	I
cfg_rcw_src2	Power-on-Reset Configuration	<b>IFC_AD10</b>	I
cfg_rcw_src3	Power-on-Reset Configuration	<b>IFC_AD11</b>	I
cfg_rcw_src4	Power-on-Reset Configuration	<b>IFC_AD12</b>	I
cfg_rcw_src5	Power-on-Reset Configuration	<b>IFC_AD13</b>	I
cfg_rcw_src6	Power-on-Reset Configuration	<b>IFC_AD14</b>	I
cfg_rcw_src7	Power-on-Reset Configuration	<b>IFC_AD15</b>	I
cfg_rcw_src8	Power-on-Reset Configuration	<b>IFC_CLE</b>	I
<b>General Purpose Input/Output</b>			
GPIO1_13	General Purpose Input/Output	<b>ASLEEP</b>	O
GPIO1_14	General Purpose Input/Output	<b>RTC</b>	IO
GPIO1_15	General Purpose Input/Output	<b>UART1_SOUT</b>	IO
GPIO1_16	General Purpose Input/Output	<b>UART2_SOUT</b> LPUART1_SOUT FTM4_CH0	IO
GPIO1_17	General Purpose Input/Output	<b>UART1_SIN</b>	IO
GPIO1_18	General Purpose Input/Output	<b>UART2_SIN</b> FTM4_CH1 LPUART1_SIN	IO
GPIO1_19	General Purpose Input/Output	<b>UART1_RTS_B</b> UART3_SOUT LPUART2_SOUT FTM4_CH2	IO
GPIO1_20	General Purpose Input/Output	<b>UART2_RTS_B</b> UART4_SOUT LPUART4_SOUT FTM4_CH3 LPUART1_RTS_B	IO
GPIO1_21	General Purpose Input/Output	<b>UART1_CTS_B</b> UART3_SIN FTM4_CH4 LPUART2_SIN	IO
GPIO1_22	General Purpose Input/Output	<b>UART2_CTS_B</b> UART4_SIN FTM4_CH5 LPUART1_CTS_B LPUART4_SIN	IO
GPIO1_23	General Purpose Input/Output	<b>IRQ03</b> FTM3_CH7 TDMB_TSYNC UC3_RTSB_TXEN	IO
GPIO1_24	General Purpose Input/Output	<b>IRQ04</b> FTM3_CH0	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMA_RXD UC1_RXD7 TDMA_TXD	
GPIO1_25	General Purpose Input/Output	IRQ05 FTM3_CH1 TDMA_RSYNC UC1_CTSB_RXDV	IO
GPIO1_26	General Purpose Input/Output	IRQ06 FTM3_CH2 TDMA_RXD_EXC TDMA_TXD UC1_TXD7	IO
GPIO1_27	General Purpose Input/Output	IRQ07 FTM3_CH3 TDMA_TSYNC UC1_RTSB_TXEN	IO
GPIO1_28	General Purpose Input/Output	IRQ08 FTM3_CH4 TDMB_RXD UC3_RXD7 TDMB_TXD	IO
GPIO1_29	General Purpose Input/Output	IRQ09 FTM3_CH5 TDMB_RSYNC UC3_CTSB_RXDV	IO
GPIO1_30	General Purpose Input/Output	IRQ10 FTM3_CH6 TDMB_RXD_EXC TDMB_TXD UC3_TXD7	IO
GPIO1_31	General Purpose Input/Output	IRQ11	IO
GPIO2_00	General Purpose Input/Output	SPI_PCS0 SDHC_DAT4 SDHC_VS	IO
GPIO2_01	General Purpose Input/Output	SPI_PCS1 SDHC_DAT5 SDHC_CMD_DIR	IO
GPIO2_02	General Purpose Input/Output	SPI_PCS2 SDHC_DAT6 SDHC_DAT0_DIR	IO
GPIO2_03	General Purpose Input/Output	SPI_PCS3 SDHC_DAT7 SDHC_DAT123_DIR	IO
GPIO2_04	General Purpose Input/Output	SDHC_CMD LPUART3_SOUT FTM4_CH6	IO
GPIO2_05	General Purpose Input/Output	SDHC_DAT0 FTM4_CH7	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		LPUART3_SIN	
GPIO2_06	General Purpose Input/Output	<b>SDHC_DAT1</b> LPUART5_SOUT FTM4_FAULT LPUART2_RTS_B	IO
GPIO2_07	General Purpose Input/Output	<b>SDHC_DAT2</b> LPUART2_CTS_B LPUART5_SIN FTM4_EXTCLK	IO
GPIO2_08	General Purpose Input/Output	<b>SDHC_DAT3</b> LPUART6_SOUT FTM4_QD_PHA LPUART3_RTS_B	IO
GPIO2_09	General Purpose Input/Output	<b>SDHC_CLK</b> LPUART3_CTS_B LPUART6_SIN FTM4_QD_PHB	IO
GPIO2_10	General Purpose Input/Output	<b>IFC_CS1_B</b> FTM7_CH0	IO
GPIO2_11	General Purpose Input/Output	<b>IFC_CS2_B</b> FTM7_CH1	IO
GPIO2_12	General Purpose Input/Output	<b>IFC_CS3_B</b> QSPI_B_DATA3 FTM7_EXTCLK	IO
GPIO2_13	General Purpose Input/Output	<b>IFC_PAR0</b> QSPI_B_DATA0 FTM6_CH0	IO
GPIO2_14	General Purpose Input/Output	<b>IFC_PAR1</b> QSPI_B_DATA1 FTM6_CH1	IO
GPIO2_15	General Purpose Input/Output	<b>IFC_PERR_B</b> QSPI_B_DATA2 FTM6_EXTCLK	IO
GPIO2_25	General Purpose Input/Output	<b>IFC_A25</b> QSPI_A_DATA3 FTM5_CH0 IFC_CS4_B IFC_RB2_B	IO
GPIO2_26	General Purpose Input/Output	<b>IFC_A26</b> FTM5_CH1 IFC_CS5_B IFC_RB3_B	IO
GPIO2_27	General Purpose Input/Output	<b>IFC_A27</b> FTM5_EXTCLK IFC_CS6_B	IO
GPIO3_00	General Purpose Input/Output	<b>EMI1_MDC</b>	IO
GPIO3_01	General Purpose Input/Output	<b>EMI1_MDIO</b>	IO
GPIO3_02	General Purpose Input/Output	<b>EC1_TXD3</b>	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		FTM1_CH5	
GPIO3_03	General Purpose Input/Output	<b>EC1_TXD2</b> FTM1_CH7	IO
GPIO3_04	General Purpose Input/Output	<b>EC1_TXD1</b> FTM1_CH3	IO
GPIO3_05	General Purpose Input/Output	<b>EC1_TXD0</b> FTM1_CH2	IO
GPIO3_06	General Purpose Input/Output	<b>EC1_TX_EN</b> FTM1_FAULT	IO
GPIO3_07	General Purpose Input/Output	<b>EC1_GTX_CLK</b> FTM1_EXTCLK	IO
GPIO3_08	General Purpose Input/Output	<b>EC1_GTX_CLK12</b> 5	IO
GPIO3_09	General Purpose Input/Output	<b>EC1_RXD3</b> FTM1_CH4	IO
GPIO3_10	General Purpose Input/Output	<b>EC1_RXD2</b> FTM1_CH6	IO
GPIO3_11	General Purpose Input/Output	<b>EC1_RXD1</b> FTM1_CH1	IO
GPIO3_12	General Purpose Input/Output	<b>EC1_RXD0</b> FTM1_CH0	IO
GPIO3_13	General Purpose Input/Output	<b>EC1_RX_CLK</b> FTM1_QD_PHA	IO
GPIO3_14	General Purpose Input/Output	<b>EC1_RX_DV</b> FTM1_QD_PHB	IO
GPIO3_15	General Purpose Input/Output	<b>EC2_TXD3</b> TSEC_1588_ALAR M_OUT2 FTM2_CH5	IO
GPIO3_16	General Purpose Input/Output	<b>EC2_TXD2</b> TSEC_1588_ALAR M_OUT1 FTM2_CH7	IO
GPIO3_17	General Purpose Input/Output	<b>EC2_TXD1</b> TSEC_1588_CLK_OUT FTM2_CH3	IO
GPIO3_18	General Purpose Input/Output	<b>EC2_TXD0</b> TSEC_1588_PULS E_OUT2 FTM2_CH2	IO
GPIO3_19	General Purpose Input/Output	<b>EC2_TX_EN</b> FTM2_FAULT	IO
GPIO3_20	General Purpose Input/Output	<b>EC2_GTX_CLK</b> FTM2_EXTCLK	IO
GPIO3_21	General Purpose Input/Output	<b>EC2_GTX_CLK12</b> 5	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
GPIO3_22	General Purpose Input/Output	<b>EC2_RXD3</b> FTM2_CH4	IO
GPIO3_23	General Purpose Input/Output	<b>EC2_RXD2</b> FTM2_CH6	IO
GPIO3_24	General Purpose Input/Output	<b>EC2_RXD1</b> TSEC_1588_PULS E_OUT1 FTM2_CH1	IO
GPIO3_25	General Purpose Input/Output	<b>EC2_RXD0</b> TSEC_1588_TRIG _IN2 FTM2_CH0	IO
GPIO3_26	General Purpose Input/Output	<b>EC2_RX_CLK</b> TSEC_1588_CLK_I N FTM2_QD_PHA	IO
GPIO3_27	General Purpose Input/Output	<b>EC2_RX_DV</b> TSEC_1588_TRIG _IN1 FTM2_QD_PHB	IO
GPIO4_00	General Purpose Input/Output	<b>EMI2_MDC</b>	IO
GPIO4_01	General Purpose Input/Output	<b>EMI2_MDIO</b>	IO
GPIO4_02	General Purpose Input/Output	<b>IIC2_SCL</b> SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0 BRGO2	IO
GPIO4_03	General Purpose Input/Output	<b>IIC2_SDA</b> SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1 BRGO3	IO
GPIO4_10	General Purpose Input/Output	<b>IIC3_SCL</b> EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0 CLK11	IO
GPIO4_11	General Purpose Input/Output	<b>IIC3_SDA</b> EVT6_B USB2_PWRFAULT BRGO1 FTM8_CH1 CLK12_CLK8	IO
GPIO4_12	General Purpose Input/Output	<b>IIC4_SCL</b> EVT7_B USB3_DRVVBUS	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMA_RQ FTM3_FAULT UC1_CDB_RXER	
GPIO4_13	General Purpose Input/Output	IIC4_SDA EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK UC3_CDB_RXER	IO
GPIO4_29	General Purpose Input/Output	USB_DRVVBUS	IO
GPIO4_30	General Purpose Input/Output	USB_PWRFAULT	IO
<b>Frequency Timer Module 1</b>			
FTM1_CH0	Channel 0	EC1_RXD0 GPIO3_12	IO
FTM1_CH1	Channel 1	EC1_RXD1 GPIO3_11	IO
FTM1_CH2	Channel 2	EC1_TXD0 GPIO3_05	IO
FTM1_CH3	Channel 3	EC1_TXD1 GPIO3_04	IO
FTM1_CH4	Channel 4	EC1_RXD3 GPIO3_09	IO
FTM1_CH5	Channel 5	EC1_TXD3 GPIO3_02	IO
FTM1_CH6	Channel 6	EC1_RXD2 GPIO3_10	IO
FTM1_CH7	Channel 7	EC1_TXD2 GPIO3_03	IO
FTM1_EXTCLK	Ext Clock	EC1_GTX_CLK GPIO3_07	I
FTM1_FAULT	Fault	EC1_TX_EN GPIO3_06	I
FTM1_QD_PHA	Phase A	EC1_RX_CLK GPIO3_13	I
FTM1_QD_PHB	Phase B	EC1_RX_DV GPIO3_14	I
<b>Frequency Timer Module 2</b>			
FTM2_CH0	Channel 0	EC2_RXD0 GPIO3_25 TSEC_1588_TRIG_IN2	IO
FTM2_CH1	Channel 1	EC2_RXD1 GPIO3_24 TSEC_1588_PULSE_OUT1	IO
FTM2_CH2	Channel 2	EC2_TXD0	IO

*Table continues on the next page...*

## Signals Overview

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		GPIO3_18 TSEC_1588_PULS E_OUT2	
FTM2_CH3	Channel 3	<b>EC2_TXD1</b> GPIO3_17 TSEC_1588_CLK_OUT	IO
FTM2_CH4	Channel 4	<b>EC2_RXD3</b> GPIO3_22	IO
FTM2_CH5	Channel 5	<b>EC2_TXD3</b> GPIO3_15 TSEC_1588_ALARM_OUT2	IO
FTM2_CH6	Channel 6	<b>EC2_RXD2</b> GPIO3_23	IO
FTM2_CH7	Channel 7	<b>EC2_TXD2</b> GPIO3_16 TSEC_1588_ALARM_OUT1	IO
FTM2_EXTCLK	Ext Clock	<b>EC2_GTX_CLK</b> GPIO3_20	I
FTM2_FAULT	Fault	<b>EC2_TX_EN</b> GPIO3_19	I
FTM2_QD_PHA	Phase A	<b>EC2_RX_CLK</b> GPIO3_26 TSEC_1588_CLK_IN	I
FTM2_QD_PHB	Phase B	<b>EC2_RX_DV</b> GPIO3_27 TSEC_1588_TRIG_IN1	I
<b>Frequency Timer Module 3</b>			
FTM3_CH0	Channel 0	<b>IRQ04</b> GPIO1_24 TDMA_RXD UC1_RXD7 TDMA_TXD	IO
FTM3_CH1	Channel 1	<b>IRQ05</b> GPIO1_25 TDMA_RSYNC UC1_CTSB_RXDV	IO
FTM3_CH2	Channel 2	<b>IRQ06</b> GPIO1_26 TDMA_RXD_EXC TDMA_TXD UC1_RXD7	IO
FTM3_CH3	Channel 3	<b>IRQ07</b> GPIO1_27 TDMA_TSYNC	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		UC1_RTSB_TXEN	
FTM3_CH4	Channel 4	IRQ08 GPIO1_28 TDMB_RXD UC3_RXD7 TDMB_TXD	IO
FTM3_CH5	Channel 5	IRQ09 GPIO1_29 TDMB_RSYNC UC3_CTSB_RXDV	IO
FTM3_CH6	Channel 6	IRQ10 GPIO1_30 TDMB_RXD_EXC TDMB_TXD UC3_RXD7	IO
FTM3_CH7	Channel 7	IRQ03 GPIO1_23 TDMB_TSYNC UC3_RTSB_TXEN	IO
FTM3_EXTCLK	Ext Clock	IIC4_SDA GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ UC3_CDB_RXER	I
FTM3_FAULT	Fault	IIC4_SCL GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ UC1_CDB_RXER	I
FTM3_QD_PHA	Phase A	IIC2_SCL GPIO4_02 SDHC_CD_B CLK9 QE_SI1_STROBE0 BRGO2	I
FTM3_QD_PHB	Phase B	IIC2_SDA GPIO4_03 SDHC_WP CLK10 QE_SI1_STROBE1 BRGO3	I
<b>Frequency Timer Module 4</b>			
FTM4_CH0	Channel 0	UART2_SOUT GPIO1_16 LPUART1_SOUT	IO
FTM4_CH1	Channel 1	UART2_SIN GPIO1_18 LPUART1_SIN	IO

*Table continues on the next page...*

## Signals Overview

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
FTM4_CH2	Channel 2	UART1_RTS_B GPIO1_19 UART3_SOUT LPUART2_SOUT	IO
FTM4_CH3	Channel 3	UART2_RTS_B GPIO1_20 UART4_SOUT LPUART4_SOUT LPUART1_RTS_B	IO
FTM4_CH4	Channel 4	UART1_CTS_B GPIO1_21 UART3_SIN LPUART2_SIN	IO
FTM4_CH5	Channel 5	UART2_CTS_B GPIO1_22 UART4_SIN LPUART1_CTS_B LPUART4_SIN	IO
FTM4_CH6	Channel 6	SDHC_CMD GPIO2_04 LPUART3_SOUT	IO
FTM4_CH7	Channel 7	SDHC_DAT0 GPIO2_05 LPUART3_SIN	IO
FTM4_EXTCLK	Ext Clock	SDHC_DAT2 GPIO2_07 LPUART2_CTS_B LPUART5_SIN	I
FTM4_FAULT	Fault	SDHC_DAT1 GPIO2_06 LPUART5_SOUT LPUART2_RTS_B	I
FTM4_QD_PHA	Phase A	SDHC_DAT3 GPIO2_08 LPUART6_SOUT LPUART3_RTS_B	I
FTM4_QD_PHB	Phase B	SDHC_CLK GPIO2_09 LPUART3_CTS_B LPUART6_SIN	I
<b>Frequency Timer Module 5</b>			
FTM5_CH0	Channel 0	IFC_A25 GPIO2_25 QSPI_A_DATA3 IFC_CS4_B IFC_RB2_B	IO
FTM5_CH1	Channel 1	IFC_A26 GPIO2_26 IFC_CS5_B	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		IFC_RB3_B	
FTM5_EXTCLK	Ext Clock	IFC_A27 GPIO2_27 IFC_CS6_B	I
<b>Frequency Timer Module 6</b>			
FTM6_CH0	Channel 0	IFC_PAR0 GPIO2_13 QSPI_B_DATA0	IO
FTM6_CH1	Channel 1	IFC_PAR1 GPIO2_14 QSPI_B_DATA1	IO
FTM6_EXTCLK	Ext Clock	IFC_PERR_B GPIO2_15 QSPI_B_DATA2	I
<b>Frequency Timer Module 7</b>			
FTM7_CH0	Channel 0	IFC_CS1_B GPIO2_10	IO
FTM7_CH1	Channel 1	IFC_CS2_B GPIO2_11	IO
FTM7_EXTCLK	Ext Clock	IFC_CS3_B GPIO2_12 QSPI_B_DATA3	I
<b>Frequency Timer Module 8</b>			
FTM8_CH0	Channel 0	IIC3_SCL GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 CLK11	IO
FTM8_CH1	Channel 1	IIC3_SDA GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1 CLK12_CLK8	IO
<b>LPUART</b>			
LPUART1_CTS_B / LPUART4_SIN	Clear to send	UART2_CTS_B GPIO1_22 UART4_SIN FTM4_CH5	I
LPUART1_RTS_B / LPUART4_SOUT	Request to send	UART2_RTS_B GPIO1_20 UART4_SOUT FTM4_CH3	O
LPUART1_SIN	Receive data	UART2_SIN GPIO1_18 FTM4_CH1	I

Table continues on the next page...

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
LPUART1_SOUT	Transmit data	UART2_SOUT GPIO1_16 FTM4_CH0	IO
LPUART2_CTS_B / LPUART5_SIN	Clear to send	SDHC_DAT2 GPIO2_07 FTM4_EXTCLK	I
LPUART2_RTS_B / LPUART5_SOUT	Request to send	SDHC_DAT1 GPIO2_06 FTM4_FAULT	O
LPUART2_SIN	Receive data	UART1_CTS_B GPIO1_21 UART3_SIN FTM4_CH4	I
LPUART2_SOUT	Transmit data	UART1_RTS_B GPIO1_19 UART3_SOUT FTM4_CH2	IO
LPUART3_CTS_B / LPUART6_SIN	Clear to send	SDHC_CLK GPIO2_09 FTM4_QD_PHB	I
LPUART3_RTS_B / LPUART6_SOUT	Request to send	SDHC_DAT3 GPIO2_08 FTM4_QD_PHA	O
LPUART3_SIN	Receive data	SDHC_DAT0 GPIO2_05 FTM4_CH7	I
LPUART3_SOUT	Transmit data	SDHC_CMD GPIO2_04 FTM4_CH6	IO
LPUART4_SIN / LPUART1_CTS_B	Receive data	UART2_CTS_B GPIO1_22 UART4_SIN FTM4_CH5	I
LPUART4_SOUT / LPUART1_RTS_B	Transmit data	UART2_RTS_B GPIO1_20 UART4_SOUT FTM4_CH3	IO
LPUART5_SIN / LPUART2_CTS_B	Receive data	SDHC_DAT2 GPIO2_07 FTM4_EXTCLK	I
LPUART5_SOUT / LPUART2_RTS_B	Transmit data	SDHC_DAT1 GPIO2_06 FTM4_FAULT	IO
LPUART6_SIN / LPUART3_CTS_B	Receive data	SDHC_CLK GPIO2_09 FTM4_QD_PHB	I
LPUART6_SOUT / LPUART3_RTS_B	Transmit data	SDHC_DAT3 GPIO2_08 FTM4_QD_PHA	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
<b>QUICC Engine</b>			
CLK10 / QE_SI1_STROBE1	QE clock	IIC2_SDA GPIO4_03 SDHC_WP FTM3_QD_PHB BRGO3	I
CLK11	QE clock	IIC3_SCL GPIO4_10 EVT5_B USB2_DRVVBUS BRGO4 FTM8_CH0	I
CLK12_CLK8	QE clock	IIC3_SDA GPIO4_11 EVT6_B USB2_PWRFAULT BRGO1 FTM8_CH1	I
CLK9 / QE_SI1_STROBE0	QE clock	IIC2_SCL GPIO4_02 SDHC_CD_B FTM3_QD_PHA BRGO2	I
QE_SI1_STROBE0 / CLK9	SI strobe	IIC2_SCL GPIO4_02 SDHC_CD_B FTM3_QD_PHA BRGO2	O
QE_SI1_STROBE1 / CLK10	SI strobe	IIC2_SDA GPIO4_03 SDHC_WP FTM3_QD_PHB BRGO3	O
UC1_CDB_RXER	Receive error	IIC4_SCL GPIO4_12 EVT7_B USB3_DRVVBUS TDMA_RQ FTM3_FAULT	I
UC1_CTSB_RXDV	Receive data	IRQ05 GPIO1_25 FTM3_CH1 TDMA_RSYNC	I
UC1_RTSB_TXEN	Transmit enable	IRQ07 GPIO1_27 FTM3_CH3 TDMA_TSYNC	O
UC1_RXD7	Receive data	IRQ04 GPIO1_24 FTM3_CH0	I

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		TDMA_RXD TDMA_TXD	
UC1_TXD7	Transmit data	<b>IRQ06</b> GPIO1_26 FTM3_CH2 TDMA_RXD_EXC TDMA_TXD	O
UC3_CDB_RXER	Receive error	<b>IIC4_SDA</b> GPIO4_13 EVT8_B USB3_PWRFAULT TDMB_RQ FTM3_EXTCLK	I
UC3_CTSB_RXDV	Receive data	<b>IRQ09</b> GPIO1_29 FTM3_CH5 TDMB_RSYNC	I
UC3_RTSB_TXEN	Transmit enable	<b>IRQ03</b> GPIO1_23 FTM3_CH7 TDMB_TSYNC	O
UC3_RXD7	Receive data	<b>IRQ08</b> GPIO1_28 FTM3_CH4 TDMB_RXD TDMB_TXD	I
UC3_TXD7	Transmit data	<b>IRQ10</b> GPIO1_30 FTM3_CH6 TDMB_RXD_EXC TDMB_TXD	O
<b>Baud rate generator</b>			
BRGO1	Baud Rate Generator 1	<b>IIC3_SDA</b> GPIO4_11 EVT6_B USB2_PWRFAULT FTM8_CH1 CLK12_CLK8	O
BRGO2	Baud Rate Generator 2	<b>IIC2_SCL</b> GPIO4_02 SDHC_CD_B FTM3_QD_PHA CLK9 QE_SI1_STROBE0	O
BRGO3	Baud Rate Generator 3	<b>IIC2_SDA</b> GPIO4_03 SDHC_WP FTM3_QD_PHB CLK10 QE_SI1_STROBE1	O

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
BRGO4	Baud Rate Generator 4	IIC3_SCL GPIO4_10 EVT5_B USB2_DRVVBUS FTM8_CH0 CLK11	O
<b>Time Division Multiplexing</b>			
TDMA_RQ	RQ	IIC4_SCL GPIO4_12 EVT7_B USB3_DRVVBUS FTM3_FAULT UC1_CDB_RXER	O
TDMA_RSYNC	RSYNC	IRQ05 GPIO1_25 FTM3_CH1 UC1_CTSB_RXDV	I
TDMA_RXD / TDMA_TXD	RXD	IRQ04 GPIO1_24 FTM3_CH0 UC1_RXD7	I
TDMA_RXD_EXC / TDMA_TXD	Recieve Data	IRQ06 GPIO1_26 FTM3_CH2 UC1_TXD7	I
TDMA_TSYNC	TSYNC	IRQ07 GPIO1_27 FTM3_CH3 UC1_RTSB_TXEN	I
TDMA_TXD / TDMA_RXD	Transmit Data	IRQ04 GPIO1_24 FTM3_CH0 UC1_RXD7	O
TDMA_TXD / TDMA_RXD_EXC	Transmit Data	IRQ06 GPIO1_26 FTM3_CH2 UC1_TXD7	O
TDMB_RQ	RQ	IIC4_SDA GPIO4_13 EVT8_B USB3_PWRFAULT FTM3_EXTCLK UC3_CDB_RXER	O
TDMB_RSYNC	RSYNC	IRQ09 GPIO1_29 FTM3_CH5 UC3_CTSB_RXDV	I
TDMB_RXD / TDMB_TXD	RXD	IRQ08 GPIO1_28 FTM3_CH4	I

*Table continues on the next page...*

## Signals Overview

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
		UC3_RXD7	
TDMB_RXD_EXC / TDMB_TXD	Recieve Data	IRQ10 GPIO1_30 FTM3_CH6 UC3_RXD7	I
TDMB_TSYNC	TSYNC	IRQ03 GPIO1_23 FTM3_CH7 UC3_RTSB_TXEN	I
TDMB_TXD / TDMB_RXD	Transmit Data	IRQ08 GPIO1_28 FTM3_CH4 UC3_RXD7	O
TDMB_TXD / TDMB_RXD_EXC	Transmit Data	IRQ10 GPIO1_30 FTM3_CH6 UC3_RXD7	O
<b>TSEC_1588</b>			
TSEC_1588_ALARM_O_UT1	Alarm Out	EC2_RXD2 GPIO3_16 FTM2_CH7	O
TSEC_1588_ALARM_O_UT2	Alarm Out	EC2_RXD3 GPIO3_15 FTM2_CH5	O
TSEC_1588_CLK_IN	Clock In	EC2_RX_CLK GPIO3_26 FTM2_QD_PHA	I
TSEC_1588_CLK_OUT	Clock Out	EC2_RXD1 GPIO3_17 FTM2_CH3	O
TSEC_1588_PULSE_OU_T1	Pulse Out	EC2_RXD1 GPIO3_24 FTM2_CH1	O
TSEC_1588_PULSE_OU_T2	Pulse Out	EC2_RXD0 GPIO3_18 FTM2_CH2	O
TSEC_1588_TRIG_IN1	Trigger In	EC2_RX_DV GPIO3_27 FTM2_QD_PHB	I
TSEC_1588_TRIG_IN2	Trigger In	EC2_RXD0 GPIO3_25 FTM2_CH0	I
<b>QSPI</b>			
QSPI_A_CS0	Chip Select	IFC_A16	O
QSPI_A_CS1	Chip Select	IFC_A17	O
QSPI_A_DATA0	Data	IFC_A22 IFC_WP1_B	IO

*Table continues on the next page...*

**Table B-1. LS1043A\_23X23 Signal Reference by Functional Block (continued)**

Name	Description	Alternate Function(s)	Pin type
QSPI_A_DATA1	Data	IFC_A23 IFC_WP2_B	IO
QSPI_A_DATA2	Data	IFC_A24 IFC_WP3_B	IO
QSPI_A_DATA3	Data	IFC_A25 GPIO2_25 FTM5_CH0 IFC_CS4_B IFC_RB2_B	IO
QSPI_A_SCK	Serial Clock	IFC_A18	O
QSPI_B_CS0	Chip Select	IFC_A19	O
QSPI_B_CS1	Chip Select	IFC_A20	O
QSPI_B_DATA0	Data	IFC_PAR0 GPIO2_13 FTM6_CH0	IO
QSPI_B_DATA1	Data	IFC_PAR1 GPIO2_14 FTM6_CH1	IO
QSPI_B_DATA2	Data	IFC_PERR_B GPIO2_15 FTM6_EXTCLK	IO
QSPI_B_DATA3	Data	IFC_CS3_B GPIO2_12 FTM7_EXTCLK	IO
QSPI_B_DATA3	Data	IFC_CS3_B GPIO2_12 FTM7_EXTCLK	IO
QSPI_B_SCK	Serial Clock	IFC_A21 cfg_dram_type	O

## B.3 LS1043A signal mapping

The following table lists the mapping for those signals whose names are different for 21x21 package and 23x23 package.

**Table B-2. LS1043A signal mapping**

DDR	
D1_MB1	D1_MA14
D1_MACT_B	D1_MA15
D1_MALERT_B	D1_MAPAR_ERR_B
D1_MPAPR	D1_MAPAR_OUT

*Table continues on the next page...*

**Table B-2. LS1043A signal mapping (continued)**

D1_MBGO	D1_MBA2
<b>SPI</b>	
PCS0	SPI_CS_B[0]
PCS[1:3]	SPI_CS_B[1:3]
SPI_SCK	SPI_CLK
SPI_SIN	SPI_MISO
SPI_SOUT	SPI_MOSI

## B.4 Register differences

The following table describes register differences for LS1043A 23x23 package from 21x21 package:

**Table B-3. Differences for LS1043A 23x23 package**

Register Name	Bit Number	Description
System Version Register (DCFG_CCSR_SVR)	16–23 (VAR_PER)	Various Personalities 0000_0010 - LS1043A 23x23 package (with security) 0000_0011 - LS1043A 23x23 package (without security) 0000_1010 - LS1023A 23x23 package (with security) 0000_1011 - LS1023A 23x23 package (without security)
PCI Express Device ID Register (Device_ID_Register)	15–0 (Device_ID)	Device ID 10000000_10000010 - LS1043A 23x23 package (with security) 10000000_10000011 - LS1043A 23x23 package (without security) 10000000_10001010 - LS1023A 23x23 package (without security) 10000000_10001011 - LS1023A 23x23 package (without security)

# Appendix C

## EPU register descriptions

All registers are accessible with 32-bit accesses. The EPU registers are configurable via DCSR.

For example, to access the EPEVTCR0 register, the following physical address must be used.

$$(\$DCSR\_BASE + \$EPU\_BASE + \text{register offset}) = 0x07\_0000\_0000 + 0x6\_0000 + 0x50 = 0x07\_0006\_0050$$

### C.1 EPU\_dcsr memory map

EPU base address: 6\_0000h

Offset	Register	Width (In bits)	Access	Reset value
50h - 74h	Event Processor EVT Pin Control Register n (EPEVTCR0 - EPEVTCR9)	32	RW	0000_0000h
FD0h - FECh	Event Processor Debug Reservation Register n (EPRSRV0 - EPRSRV7)	32	RW	0000_0000h

### C.2 Event Processor EVT Pin Control Register n (EPEVTCR0 - EPEVTCR9)

#### C.2.1 Offset

For a = 0 to 9:

## Event Processor EVT Pin Control Register n (EPEVTCR0 - EPEVTCR9)

Register	Offset
EPEVTCRa	50h + (a × 4h)

### C.2.2 Function

The SoC's EVT pins are muxed among different functionalities. For each EVT pin, a control bit is set within this register to indicate the input or output direction. A value of zero means the pin is an input to EPU. A value of one means the pin is an output of EPU.

### C.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EVT_SEL				0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DIR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### C.2.4 Fields

Field	Function
31-28 EVT_SEL	Event Select. Value determines which event is mapped to this EVT pin when it is configured as an output  0000b - SCU0 drives EVTn when DIR=1 0001b - SCU1 drives EVTn when DIR=1 0010b - SCU2 drives EVTn when DIR=1 0011b - SCU3 drives EVTn when DIR=1 0100b - SCU4 drives EVTn when DIR=1 0101b - SCU5 drives EVTn when DIR=1 0110b - SCU6 drives EVTn when DIR=1 0111b - SCU7 drives EVTn when DIR=1 1000b - SCU8 drives EVTn when DIR=1 1001b - SCU9 drives EVTn when DIR=1 1010b - SCU10 drives EVTn when DIR=1 1011b - SCU11 drives EVTn when DIR=1 1100b - SCU12 drives EVTn when DIR=1 1101b - SCU13 drives EVTn when DIR=1 1110b - SCU14 drives EVTn when DIR=1 1111b - SCU15 drives EVTn when DIR=1

Table continues on the next page...

Field	Function
27-1 —	Reserved
0 DIR	Direction 0b - = EVTn is a device input 1b - = EVTn is a device output driven by EPU

## C.3 Event Processor Debug Reservation Register n (EPRS RV0 - EPRSRV7)

### C.3.1 Offset

For  $a = 0$  to 7:

Register	Offset
EPRSRVa	FECh + ( $a \times -4h$ )

### C.3.2 Function

The debug reservation assist registers consist of a set of hardware semaphores and a bank of read/write scratch registers (reservation block). The intent is that these can be used to facilitate the implementation of a resource sharing protocol for device debug resources among the various tools and agents that are likely to want access to them.

The debug reservation block is a bank of scratch registers with read/write access which are intended for use in developing a resource sharing protocol. The nature of the protocol is at the discretion of software and is not enforced in any way by hardware.

### C.3.3 Diagram

Bits	31	30	29	28	27	26	25	24		23	22	21	20	19	18	17	16
R									SCRATCH								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
Bits	15	14	13	12	11	10	9	8		7	6	5	4	3	2	1	0
R									SCRATCH								
W																	
Reset	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0

### C.3.4 Fields

Field	Function
31-0 SCRATCH	Scratch space available for implementing a resource sharing protocol.

# **Appendix D**

## **Appendix LS1043A AEC-Q100 qualified versions**

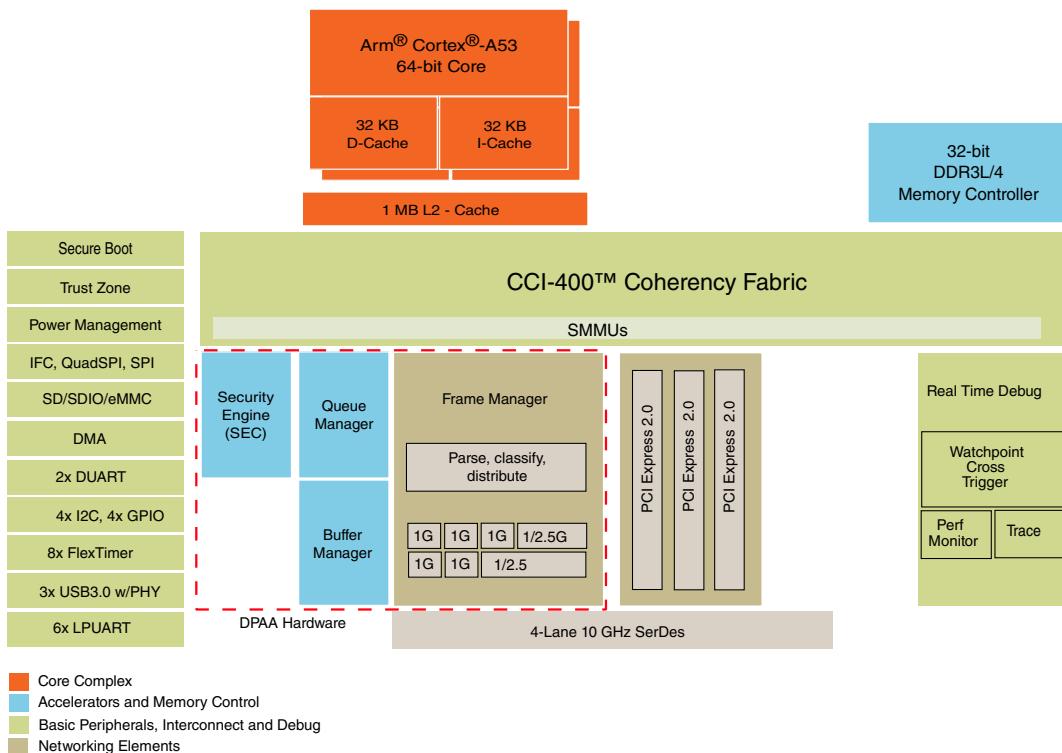
### **D.1 Introduction**

LS1043A AEC-Q100 is a cost-effective, power-efficient, and highly integrated system-on-chip (SoC) design that extends the reach of the NXP value-performance line of QorIQ communications processors. Featuring extremely power-efficient 64-bit Arm® Cortex®-A53 cores with ECC-protected L1 and L2 cache memories for high reliability, running up to 1.6 GHz.

LS1043A AEC-Q100 is ideal for applications such as, advanced gateways, vehicle network processing, service-oriented gateways.

The following figure shows the major functional units within the LS1043A AEC-Q100 part.

## Differences between LS1043A'C' grade 3 and LS1043A'B' grade 2



**Figure D-1. Block diagram**

LS1043A'C' is the root part number for the grade 3 qualified version of the product, which supports complete feature set of the LS1043A. The following sections describes the module differences between LS1043A'C', and the grade 2 qualified version, with root part number LS1043A'B'.

## D.2 Differences between LS1043A'C' grade 3 and LS1043A'B' grade 2

The following table describes the module differences between LS1043A'C' grade 3 and LS1043A'B' grade 2.

**Table D-1. Differences between LS1043A'C' grade 3 and LS1043A'B' grade 2**

Features	LS1043A'C' grade 3	LS1043A'B' grade 2
Arm Cortex-A53 Processor speed	Up to 1.6 GHz	Up to 1.4 GHz
Hierarchical interconnect fabric frequency	Up to 400 MHz	Up to 300 MHz
SATA controller	Yes	No
FMan Ethernet XFI (10 GbE) interface	Yes	No
FMan Ethernet QSGMII	Yes	No
uQE supporting TDM/HDLC	Yes	No

*Table continues on the next page...*

**Table D-1. Differences between LS1043A'C' grade 3 and LS1043A'B' grade 2 (continued)**

Features	LS1043A'C' grade 3	LS1043A'B' grade 2
Top SerDes frequency	10 GHz	5 GHz

## D.3 CCSR address map

The following table lists the CCSR address map changes applicable to LS1043A'B' grade 2.

**Table D-2. CCSR block base address map**

Block Base Address (Hex)	Module Name
240_0000-27F_FFFF (QUICC engine)	Reserved
320_0000-320_FFFF (SATA)	Reserved

## D.4 RCW field definition differences

The following table lists the RCW field definition differences applicable to LS1043A'B' grade 2.

**Table D-3. RCW field definition differences**

Fields(s) (of 0-511)	Field Name	Description	Notes/comments
166-167	Reserved		
357-359	IRQ_EXT	This field configures the functionality of IRQ[3:11] pins together with IRQ_BASE field.	Options: 000 See IRQ_BASE field definition 011 {FTM3_CH7, FTM3_CH0, FTM3_CH1, FTM3_CH2, FTM3_CH3, FTM3_CH4, FTM3_CH5, FTM3_CH6, GPIO1_31} Settings not shown are reserved.
445-447	IIC2_EXT	Selects between IIC2 base functionality and extension.	Options: 000 IIC2_SCL, IIC2_SDA 001 SDHC_CD_B, SDHC_WP 010 GPIO4_2, GPIO4_3 011 FTM3_QD_PHA, FTM3_QD_PHB Settings not shown are reserved.
437	Reserved		

## D.5 Internal interrupt sources

The following interrupts are disabled for LS1043A'B' grade 2.

**Table D-4. Interrupt differences between LS1043A'C' grade 3 and LS1043A'B' grade 2**

Internal interrupt	Interrupt source
101	Reserved
109	Reserved

## D.6 SerDes module

### D.6.1 SerDes lane assignments and multiplexing

This section describes the SerDes module lane assignment and multiplexing applicable to LS1043A'B' grade 2.

SRDS_PRTCL_S1 RCW[128:143]	A	B	C	D	PLL Mapping
0	Unused				
2355	sg.m9 (2.5G)	sg.m2	PCIe2 (x1)	PCIe3 (x1)	1222
3335	sg.m9	sg.m2	sg.m5	PCIe3 (x1)	1111
3355	sg.m9	sg.m2	PCIe2 (x1)	PCIe3 (x1)	1111
3555	sg.m9	PCIe1 (x1)	PCIe2 (x1)	PCIe3 (x1)	1111
7000	PCIe1 (x4)				1111
3333	sg.m9	sg.m2	sg.m5	sg.m6	1111
9960	PCIe1 (x1)	PCIe2 (x1)	PCIe3 (x2)		2222
2233	sg.m9 (2.5G)	sg.m2 (2.5)	sg.m5	sg.m6	1122

### D.6.2 Valid reference clocks and PLL configurations for SerDes protocols

Each supported SerDes protocol allows for a finite set of valid SerDes related RCW fields and reference clock frequencies, as shown in the following table:

**Table D-5. Valid SerDes RCW encodings and reference clocks for LS1043A'B' grade 2**

SerDes protocol (given lane)	Valid reference clock frequency	Valid setting for SRDS_PRTCL_Sn	Valid setting for SRDS_PLL_REF _CLK_SEL_Sn		Valid setting for SRDS_DIV_*_Sn
			PLL1	PLL2	
PCI Express 2.5 Gbps (doesn't negotiate upwards)	100 MHz	Any PCIe	0: 100 MHz	0: 100 MHz	10: 2.5G
	125 MHz		1: 125 MHz	1: 125 MHz	
PCI Express 5 Gbps (can negotiate up to 5 Gbps)	100 MHz	Any PCIe	0: 100 MHz	0: 100 MHz	01: 5G
	125 MHz		1: 125 MHz	1: 125 MHz	
SGMII (1.25 Gbps)	100 MHz	SGMII @ 1.25 Gbps	0: 100 MHz	0: 100 MHz	Don't Care
	125 MHz		1: 125 MHz	1: 125 MHz	
2.5 G SGMII (3.125 Gbps)	125 Mhz	SGMII @ 3.125 Gbps	0: 125 MHz	-	Don't Care
	156.25 MHz		1: 156.25 MHz	-	

Notes:

1) A spread-spectrum reference clock is permitted for PCI Express. However, if any other high speed interface such as SGMII is used concurrently on the same SerDes bank, spread-spectrum clocking is not permitted. Only 0x7000 option supports spread-spectrum clock.

2) The clock for 2.5 G SGMII can only be sourced from SerDes PLL1. Selecting a valid SerDes protocol automatically configures respective PLLs. However, a valid reference clock of either 100 MHz or 125 MHz is also required on reference input for PLL2 if the remaining SerDes lanes are used.

### D.6.3 Frame manager MACs

Each FMan supports seven MACs. These MAC's support different protocols as summarized in the table below.

**Table D-6. MAC Capabilities**

MAC	RGMII 1 Gbps	SGMII 1 Gbps	SGMII 2.5 Gbps
1	-	Y	-
2	-	Y	Y
3	Y	-	-
4	Y	-	-
5	-	Y	-
6	-	Y	-
9	-	Y	Y

Notes:

RGMII Interfaces: MAC3 and MAC4 are used for EC1 and EC2 interfaces respectively.

## D.7 Register differences

The following table describes the register differences between LS1043A'C' grade 3 and LS1043A'B' grade 2:

**Table D-7. Register differences between LS1043A'C' grade 3 and LS1043A'B' grade 2**

Register Name	Bit Number	Description
Device Disable Register 1 (DCFG_CCSR_DEVDISR1)	16 (SATA) 31 (QE)	Reserved
SATA ICID Register (SCFG_SATA_ICID)	-	Reserved
QE ICID Register (SCFG_QE_ICID)	-	Reserved
QOS2 Register (SCFG_QOS2)	24-27 (QE_QoS) 28-31 (SATA_QoS)	Reserved
Snoop Configuration Register (SCFG_SNPCNFGCR)	8 (SATARD SNP) 9 (SATAWR SNP)	Reserved
Extended RCW PinMux Control Register (SCFG_RCWPMUXCR0)	17-19 (IIC3_SCL) Setting 100 and 110 21-23 (IIC3_SDA) Setting 100 and 110 25-27 (IIC4_SCL) Setting 100 and 110 29-31 (IIC4_SDA) Setting 100 and 110	Reserved

# Appendix E

## Terminology, conventions, and resources

### E.1 About this content

The primary objective of this document is to define the functionality of the chip. The LS1043A provides integration of processing power for networking and communications peripherals, resulting in higher device performance. It contains a four Arm® Cortex®-v8 A53 processor cores each with 32KB of Instruction and data L1 cache, as well as 1 MB unified I/D L2 Cache.

#### NOTE

The diagrams in this content are provided to aid in understanding the overall functionality and features of this product. They do not depict the implementation details of the product, which are subject to change.

### E.2 Audience

It is assumed that the reader understands operating systems, microprocessor system design, and the basic principles of RISC processing.

### E.3 Acronyms and abbreviations

This table describes commonly-used acronyms and abbreviations used in this document.

**Table E-1. Acronyms and abbreviations**

Acronym/ Abbreviation	Meaning
8b/10b	8-bit/10-bit encoding
AIC	Antenna interface controller

*Table continues on the next page...*

## Acronyms and abbreviations

**Table E-1. Acronyms and abbreviations (continued)**

Acronym/ Abbreviation	Meaning
AMC	Advanced mezzanine card
ATM	Asynchronous transfer mode
ATMU	Address translation and mapping unit
BD	Buffer descriptor
BTB	Branch target buffer
BUID	Bus unit ID
CAM	Content-addressable memory
CCB	Core complex bus
CCSR	Configuration control and status register
CLASS	Chip-level arbitration and switching system
CRC	Cyclic redundancy check
DDR	Double data-rate
DIP	Dual inline package
DPLL	Digital phase-locked loop
DTLB	Data translation lookaside buffer
dTSEC	Data path three-speed Ethernet controller
DUART	Dual universal asynchronous receiver/transmitter
ECC	Error checking and correction
ECM	e500 coherency module
EEST	Enhanced Ethernet serial transceiver
EHCI	Enhanced host controller interface
eLBC	Enhanced local bus controller
EPROM	Erasable programmable read-only memory
EEPROM	Electrically-erasable programmable read-only memory
eTSEC	Enhanced three-speed Ethernet controller
FCS	Frame-check sequence
FIFO	First in, first out
GCI	General circuit interface
GMII	Gigabit media-independent interface
GPCM	General-purpose chip-select machine
GPIO	General-purpose I/O
GPR	General-purpose register
ICID	Isolation context identifier, maps to Stream ID of SMMU
I2C	Inter-integrated circuit
IFC	Intergrated Flash controller
IPG	Interpacket gap
IrDA	Infrared Data Association
ISDN	Integrated services digital network
ITLB	Instruction translation lookaside buffer

*Table continues on the next page...*

**Table E-1. Acronyms and abbreviations (continued)**

<b>Acronym/ Abbreviation</b>	<b>Meaning</b>
IU	Integer unit
HSSI	High-speed serial interface
LAE	Local access error
LAW	Local access window
LBC	Local bus controller
LIFO	Last-in-first-out
LRU	Least recently used
LSB	Least significant byte
lsb	Least significant bit
LSU	Load/store unit
MAC	Multiply accumulate, media access control
MDI	Medium-dependent interface
MII	Media independent interface
MMU	Memory management unit
MSB	Most significant byte
msb	Most significant bit
NMI	Non-maskable interrupt
NMSI	Nonmultiplexed serial interface
No-op	No operation
OCeaN	On-chip network
OC	
OSI	Open systems interconnection
PCS	Physical coding sublayer
PIC	Programmable interrupt controller
PMA	Physical medium attachment
PMD	Physical medium dependent
PLL	Phase-locked loop
POR	Power-on reset
PRI	Primary rate interface
PWM	Pulse-width modulation
RAID	Redundant array of independent drives
RGMII	Reduced gigabit media-independent interface
RIO	RapidIO
RTOS	Real-time operating system
RWITM	Read with intent to modify
RMW	Read-modify-write
Rx	Receiver
RxBD	Receive buffer descriptor
SATA	Serial advanced technology attachment

*Table continues on the next page...*

**Table E-1. Acronyms and abbreviations (continued)**

Acronym/ Abbreviation	Meaning
SCP	Serial control port
SDLC	Synchronous data link control
SDMA	Serial DMA
SD/MMC	Secure digital/multimedia card
SDOS	SmartDSP operating system
SEC	Security Engine
SerDes	Serializer/Deserializer
SFD	Start frame delimiter
SI	Serial interface
SIU	System interface unit
SMC	Serial management controller
SPI	Serial peripheral interface
SPR	Special-purpose register
SRAM	Static random access memory
TAP	Test access port
TBI	Ten-bit interface
TDM	Time-division multiplexed
TLB	Translation lookaside buffer
TSA	Time-slot assigner
Tx	Transmitter
TxBD	Transmit buffer descriptor
UART	Universal asynchronous receiver/transmitter
UPM	User-programmable machine
uTCA	Micro telecommunications computing platform
UTP	Unshielded twisted pair
VA	Virtual address
ZBT	Zero bus turnaround

## E.4 Notational conventions

This table shows notational conventions used in this content.

**Table E-2. Notational conventions**

Convention	Definition
General	
Cleared	When a bit takes the value zero, it is said to be cleared.

*Table continues on the next page...*

**Table E-2. Notational conventions (continued)**

Convention	Definition
Set	When a bit takes the value one, it is said to be set.
<b>mnemonics</b>	Instruction mnemonics are shown in lowercase bold.
<i>italics</i>	Italics can indicate the following: <ul style="list-style-type: none"> <li>Variable command parameters, for example, <b>bcctrx</b></li> <li>Titles of publications</li> <li>Internal signals, for example, <i>core int</i></li> </ul>
0x	Prefix to denote hexadecimal number
h	Suffix to denote hexadecimal number
0b	Prefix to denote binary number
b	Suffix to denote binary number
rA, rB	Instruction syntax used to identify a source GPR
rD	Instruction syntax used to identify a destination GPR
REGISTER[FIELD]	Abbreviations for registers are shown in uppercase text. Specific bits, fields, or ranges appear in brackets. For example, MSR[LE] refers to the little-endian mode enable bit in the machine state register.
x	In some contexts, such as signal encodings, an unitalicized x indicates a don't care.
<i>x</i>	An italicized x indicates an alphanumeric variable
<i>n</i>	An italicized n indicates either: <ul style="list-style-type: none"> <li>An integer variable</li> <li>A general-purpose bitfield unknown</li> </ul>
$\neg$	NOT logical operator
&	AND logical operator
	OR logical operator
	Concatenation, for example, TCR[WPEXT]    TCR[WP]
Signals	
OVERBAR	An overbar indicates that a signal is active-low.
<i>lowercase_italics</i>	Lowercase italics is used to indicate internal signals
<i>lowercase_plaintext</i>	Lowercase plain text is used to indicate signals that are used for configuration.
Register access	
Reserved	Ignored for the purposes of determining access type
R/W	Indicates that all non-reserved fields in a register are read/write
R	Indicates that all non-reserved fields in a register are read only
W	Indicates that all non-reserved fields in a register are write only
w1c	Indicates that all non-reserved fields in a register are cleared by writing ones to them

## E.5 Related resources

This table shows related resources that may be helpful. Additional literature is published as new processors become available. For current literature, visit [www.nxp.com](http://www.nxp.com) or contact your local FAE.

**Table E-3. Related resources**

Resource	Purpose
Data Sheet	Provides specific data regarding bus timing, signal behavior, and AC, DC, and thermal characteristics, as well as other design considerations
DPAA reference manual	Provides specific data regarding data path acceleration architecture including its implementation in the chip.
SEC reference manual	Provides specific data regarding the security engine implementation in the chip.
QEIWWRM	Provides specific data regarding the QUICC Engine block to implement the QUICC Engine functionality in the chip.
Application note	Addresses specific design issues useful to programmers and engineers working with NXP processors

# Appendix F

## Revision History

### F.1 Substantive changes from revision 5 to revision 6

Substantive changes from revision 5 to revision 6 are as follows:

#### F.1.1 Overview Revision History

Reference	Description
Universal Serial Bus (USB) controllers and PHY	Updated the key features list item of the USB controller from "Six programmable USB endpoints" to "Four bidirectional programmable USB endpoints".

#### F.1.2 Memory Map Revision History

Reference	Description
CCSR address map	<ul style="list-style-type: none"><li>Updated the qDMA register endianness to big-endian.</li><li>Updated the USB PHY register endianness to big-endian.</li></ul>

#### F.1.3 Signal Descriptions Revision History

Reference	Description
UART, GPIO, FTM, and LPUART signal multiplexing	<ul style="list-style-type: none"><li>Updated the RCW[UART_EXT] setting for UART1_SOUT from 000 to 0xx.</li><li>Removed the row for the UART1_SIN signal with RCW[UART_BASE]=Don't care.</li><li>Updated the RCW[UART_EXT] settings to 000 for the GPIO1[22], UART2_CTS_B, and UART4_SIN signals.</li></ul>

## F.1.4 Reset, Clocking, and Initialization Revision History

Reference	Description
<a href="#">RCW field definitions</a>	Removed the note from the IFC_GRP_E1_EXT bit.
<a href="#">IP logic clock distribution and configuration</a>	Updated the selection line for USB PHY from RCW[USB3_REFCLK_SEL] to SCFG_USB_REFCLK_SELCRn in the <a href="#">Figure 4-7</a> .

## F.1.5 Interrupt Assignments Revision History

No substantive changes

## F.1.6 ARM Modules Revision History

No substantive changes

## F.1.7 CSU, OCRAM, and MSCM Revision History

No substantive changes

## F.1.8 System Counter Revision History

No substantive changes

## F.1.9 Interconnect Fabric Revision History

No substantive changes

## F.1.10 CCI-400 Revision History

No substantive changes

## F.1.11 TZC-380 Revision History

No substantive changes

## F.1.12 Supplemental Configuration Unit Revision History

Reference	Description
Core Boot Control Register (SCFG_COREBCR)	Updated the reset value from 0000_0000h to FFFF_FFFFh.

## F.1.13 Device Configuration and Pin Control Revision History

No substantive changes

## F.1.14 Run Control and Power Management (RCPM) Revision History

No substantive changes

## F.1.15 QUICC Engine Overview Revision History

No substantive changes

## F.1.16 DPAA Overview Revision History

No substantive changes

## F.1.17 Secure Boot and Trust Architecture Revision History

Reference	Description
Non-claims	Removed the note from step 3.

## F.1.18 DDR Revision History

Reference	Description
Chip select a configuration (CS0_CONFIG - CS3_CONFIG)	Updated the CSn_CONFIG[BA_BITS_CS] bit description for 01b setting from reserved to 3 logical bank bits.

## F.1.19 Direct Memory Access Multiplexer (DMAMUX) Revision History

No substantive changes

## F.1.20 DUART Revision History

Reference	Description
UART FIFO control register (UFCR1 - UFCR2)	Updated bit 3 as reserved.

## F.1.21 Enhanced Direct Memory Access (eDMA) Revision History

No substantive changes

## F.1.22 eSDHC Revision History

Reference	Description
Clocking constraints	Added the clocking constraints section.
Tuning block procedure	Added the following note: Tuning data timeout occur with default value of SYSCTL[DTOCV] when SD card send tuning daResolution w.r.t. MC WDT reset request functional supporta after 450 $\mu$ s. According to SD host controller specification a card may take up to 150 ms (or at least 100 ms) to respond to the first tuning block.

## F.1.23 FlexTimer Module (FTM) Revision History

No substantive changes

## F.1.24 General Purpose I/O (GPIO) Revision History

No substantive changes

## F.1.25 Integrated Flash Controller (IFC) Revision History

Reference	Description
Clock Control register (IFC_CCR)	In the IFC_CCR[CLKDIV] bit description, updated the following bit settings: <ul style="list-style-type: none"> <li>• 0101: Divide by 6</li> <li>• 1011: Divide by 12</li> <li>• 1111: Divide by 16</li> </ul>
Flash Timing register 0 for Chip Select $n$ - NAND flash asyncNVDDR mode (IFC_FTIM0_CS $n$ _NAND)	Updated the register description <a href="#">Table 25-4</a> for default values of FTIMnCS0 during NOR boot.
Flash Timing register 0 for Chip Select $n$ - NAND flash Asynchronous Mode (IFC_FTIM0_CS $n$ _NAND_ASYNC_MODE)	Added the following registers: IFC_FTIM0_CS1_NAND_ASYNC_MODE to IFC_FTIM0_CS6_NAND_ASYNC_MODE.
NAND asynchronous mode boot mechanism	Added the section.
NAND asynchronous mode boot process	Added the section.

## F.1.26 Inter-Integrated Circuit (I2C) Revision History

No substantive changes

## F.1.27 LPUART Revision History

No substantive changes

## F.1.28 PCI Express Revision History

Reference	Description
PCI Express Device Capabilities Register (Device_Capabilities_Register)	Updated the FLRC bit description.
PCI Express Class Code Register (programming interface) (Class_Code_Register_a)	Renamed the class code registers and updated reset values for programming interface and base class.

*Table continues on the next page...*

## Substantive changes from revision 5 to revision 6

Reference	Description
PCI Express Class Code Register (sub class) (Class_Code_Register_b)	
PCI Express Class Code Register (base class) (Class_Code_Register_c)	
PCI Express Link Capabilities Register (Link_Capabilities_Register)	Updated the reset value of the Link_Capabilities_Register from 0x0073_F483 to 0x0073_F443.

## F.1.29 Pre-Boot Loader Revision History

No substantive changes

## F.1.30 Quad Serial Peripheral Interface (QuadSPI) Revision History

No substantive changes

## F.1.31 qDMA Revision History

Reference	Description
qDMA register descriptions	Updated the register endianness to big-endian.

## F.1.32 SATA3.0 Revision History

Reference	Description
SATA AHCI register descriptions	Added the description in each register.

## F.1.33 SerDes Module Revision History

Reference	Description
SerDes protocols	<ul style="list-style-type: none"><li>Added configuration for the SerDes protocol 2260 in the Table 33-1.</li></ul>

*Table continues on the next page...*

Reference	Description
	<ul style="list-style-type: none"> <li>Updated the Lane B configuration to PCIe 1 (x1) for the SerDes protocol 3560 in the <a href="#">Table 33-1</a>.</li> <li>Updated the PLL mapping to 1111 for the SerDes protocol 9960 in the <a href="#">Table 33-1</a>.</li> </ul>
SerDes PLLa Control Register 1 (PLL1CR1 - PLL2CR1)	Updated the reset value of PLLnCR1 from 0800_0000 to 0800_4100.
SerDes PLLa Reset Control Register (PLL1 RSTCTL - PLL2RSTCTL)	<ul style="list-style-type: none"> <li>Updated the reset value to 0467_452Fh.</li> <li>Updated the register description.</li> </ul>
Speed Switch Control Register 0 - Lane a (LNAS SCR0 - LNDSSCR0)	<ul style="list-style-type: none"> <li>Updated the OSETOVD6_0, TEQ_TYPE_0, REIDL_TH_0, and REIDL_EX_SEL_0 bit descriptions to remove configuration for "Others".</li> <li>Updated the reset value to 9828_2B00h.</li> <li>Added the statement "It is reserved for other protocols" in the register description.</li> <li>Added the ISLEW_RCTL_0 and OSLEW_RCTL_0 bits.</li> </ul>
SerDes PLLa Control Register 0 (PLL1CR0 - PLL2CR0)	Updated the PLLnCR0[DLYDIV_SEL] bit description.
Lane a Protocol Select Status Register 0 (LNAP SSR0 - LNDPSSR0)	Updated the LNnPSSR0[TYPE] bit description.
Speed Switch Control Register 1- Lane 0 (LNAS SCR1 - LNDSSCR1)	<ul style="list-style-type: none"> <li>Added the ISLEW_RCTL_1 and OSLEW_RCTL_1 bits.</li> <li>Added the statement "It is reserved for other protocols" in the register description.</li> </ul>
SGMIIa Protocol Control Register 1 (SGMIIACR1 - SGMIIDCR1)	In the SGMIIInCR1[MDEV_PORT] bit description, updated MDIO_CTL[PHY_ADDR] to MDIO_CTL[PORT_ADDR].
KX PCS Interrupt Mask (KX_VND_INT_MSK)	Removed the following setting from PCS_LNK_LOS_EN bit description: 0b - A PCS loss of link synchronization will cause an interrupt event" from the bit "PCS_LNK_LOS_EN.
1000Base-KX	Updated the step 4 for the initialization of 1000Base-KX (Clause 45) AN Advertisement Register 1.

## F.1.34 Serial Peripheral Interface Revision History

Reference	Description
Status Register (SPI_SR)	Updated the SR[TFFF] bit description to add the following statement: If FIFO is filled manually, and not by DMA see Transmit FIFO Fill Interrupt or DMA Request.

## F.1.35 Thermal Management Unit Revision History

Reference	Description
Features	Updated the temperature measurement range from 110°C to 125°C.

*Table continues on the next page...*

## Substantive changes from revision 5 to revision 6

Reference	Description
TMU register descriptions	Updated the sensor temperature measurement range from: <ul style="list-style-type: none"><li>• 110°C to 125°C</li><li>• 111°C to 126°C</li></ul>
Initialization Information	Updated TMU_TTR3CR=0x00070062 in the step 2 of programming the calibration table.

## F.1.36 Universal Serial Bus 3.0 Interface Revision History

Reference	Description
External Signals	Updated the I/O type of the USBn_VBUS signal from IO (input/output) to I (input).
USB_PHY_SS	Updated the register endianness to big-endian.

## F.1.37 WDOG Revision History

No substantive changes

## F.1.38 LS1023A-A Dual Core Version Revision History

No substantive changes

## F.1.39 Appendix LS1043A (23x23 package) Revision History

No substantive changes

## F.1.40 Event Processing Unit Revision History

Reference	Description
EPU register descriptions	Updated the description to add the configuration information.
EPU register descriptions	Removed the event processor crosstrigger control register (EPXTRIGCR).
EPU_dcsr memory map	Updated the base address to 6_0000h.
Event Processor Debug Reservation Register n (EPRSRV0 - EPRSRV7)	Updated the register description.

## F.2 Substantive changes from revision 4 to revision 5

Substantive changes from revision 4 to revision 5 are as follows:

### F.2.1 Overview Revision History

No substantive changes

### F.2.2 Memory Map Revision History

Reference	Description
<a href="#">Secure_system_counter</a> and <a href="#">Non_Secure_SYS_Counter</a>	Updated the endianness of registers.

### F.2.3 Signals Revision History

No substantive changes

### F.2.4 Reset Clocking and Initialization Revision History

Reference	Description
<a href="#">Core cluster n clock control/status register (Clocking_CLKCCSR)</a>	Updated the bit setting for 0110 for CLKCCSR[CLKSEL].
<a href="#">RCW field definitions</a>	Updated the field name CGA_PLL2_SPD[245] in the table "RCW Field Descriptions".
<a href="#">Hard-coded RCW options</a>	Updated SYS_PLL_RAT for (0x9E).
<a href="#">Table 4-9</a>	Added note 2 as table footnote.
<a href="#">System control signals</a> and <a href="#">External signal descriptions</a>	Removed sentence saying "For reset assertion to the chip, use only PORESET_B."

### F.2.5 Interrupt Assignment Revision History

No substantive changes

## F.2.6 Arm Modules Revision History

No substantive changes

## F.2.7 CSU, OCRAM, and MSCM Revision History

No substantive changes

## F.2.8 System Counter Revision History

Reference	Description
<a href="#">Secure_system_counter</a> and <a href="#">Non_Secure_SYS_Counter</a>	Updated the endianness of registers.

## F.2.9 Interconnect Fabric Revision History

No substantive changes

## F.2.10 CCI-400 Revision History

No substantive changes

## F.2.11 TrustZone Address Space Controller (TZC-380) Revision History

No substantive changes

## F.2.12 Supplemental Configuration Unit Revision History

Reference	Description
<a href="#">SCFG</a>	Updated the bit setting for SCFG_G0MSIIR[SRS].

## F.2.13 Device Configuration and Pin Control Revision History

Reference	Description
<a href="#">DDR Clock Disable Register (DCFG_CCSR_DDRCLKDR)</a>	Removed the duplicate instance of DDR Clock Disable Register.

## F.2.14 RCPM Revision History

No substantive changes

## F.2.15 QUICC Engine Block Revision History

Reference	Description
throughout the chapter	Removed the instances of Asynchronous HDLC with UART mode as only UART mode is supported.

## F.2.16 Data Path Acceleration Architecture (DPAA) Overview Revision History

No substantive changes

## F.2.17 Secure Boot and Trust Architecture Revision History

No substantive changes

## F.2.18 DDR Revision History

Reference	Description
<a href="#">DDR Control Driver Register 1 (DDRCDR_1)</a> and <a href="#">DDR Control Driver Register 2 (DDRCDR_2)</a>	Updated the reset value as 0000_0000.

## F.2.19 Direct Memory Access Multiplexer (DMAMUX) Revision History

Reference	Description
Table 19-5	Updated the table.

## F.2.20 DUART Revision History

No substantive changes

## F.2.21 Enhanced Direct Memory Access (eDMA) Revision History

No substantive changes

## F.2.22 eSDHC Revision History

No substantive changes

## F.2.23 FlexTimer Module (FTM) Revision History

No substantive changes

## F.2.24 GPIO Revision History

No substantive changes

## F.2.25 Integrated Flash Controller (IFC) Revision History

Reference	Description
Internal connectivity of WP signal and Internal connectivity of RB signal	Updated the figures to clarify IFC internal signal names and SoC signal names.

## F.2.26 I2C Revision History

Reference	Description
Divider and hold values	Updated the section and added more clarity on MUL.
Timing definitions	Revised timing definitions.

## F.2.27 Low Power Universal asynchronous receiver/ transmitter (LPUART) Revision History

No substantive changes

## F.2.28 PCI Express Interface Controller Revision History

Reference	Description
PCI Express MSI implementation	Updated the section by removing the example provided.
PEX register descriptions	Updated the 24-bit Class_Code_Register to separate it into 3 8-bit registers (offsets 0x9, 0xA, and 0xB).

## F.2.29 Pre-Boot Loader (PBL) Revision History

No substantive changes

## F.2.30 Quad Serial Peripheral Interface (QuadSPI) Revision History

Reference	Description
QuadSPI boot initialization sequence	Updated last paragraph.

## F.2.31 QDMA Revision History

No substantive changes

## F.2.32 SATA Revision History

No substantive changes

## F.2.33 SerDes Revision History

Reference	Description
SerDes protocols	Added a new protocol 3360.

## F.2.34 Serial Peripheral Interface (SPI) Revision History

Reference	Description
Serial Peripheral Interface SPI configuration	Changed 'MCR is 0b00' to 'MCR is 2b00'.
Functional description	Changed 'SPIx_MCR is 0b00' to 'SPIx_MCR is 2b00'.
throughout	Removed sections related to Peripheral Chip Select enable (PCS4/PCS5).
Memory Map/Register Definition	Added a note saying "fP and fsys are used interchangeably in this chapter."

## F.2.35 Thermal Monitor Unit Revision History

No substantive changes

## F.2.36 USB Revision History

Reference	Description
SUP_IDCODE_LO (USB_PHY_SS_IP_IDCODE_LO) and SUP_IDCODE_HI (USB_PHY_SS_SUP_IDCODE_HI)	Updated description of [DATA] bit.

## F.2.37 WDOG Revision History

Reference	Description
Watchdog reset generation	Removed the section "Reset (HRESET_B)" and moved the WDOG timeout/reset generation figure to Watchdog reset generation section.

## F.2.38 EPU Appendix

Reference	Description
throughout	Added Appendix.

## F.2.39 LS1043A\_Auto Appendix

Reference	Description
throughout	Added Appendix.

## F.3 Substantive changes from revision 3 to revision 4

Substantive changes from revision 3 to revision 4 are as follows:

### F.3.1 Overview Revision History

No substantive changes

### F.3.2 Memory Map Revision History

Reference	Description
CCSR address map	Updated endianness of CSU module.

### F.3.3 Signals Revision History

Reference	Description
Signals overview	Updated description of CLK10, CLK11, CLK12_CLK8, and CLK9 in the "QUICC Engine" section.
External IRQ, QE, and GPIO1 signal multiplexing	Updated IRQ[11] for RCW[IRQ_EXT].
SPI, eSDHC, USB and GPIO2 signal multiplexing	Updated the table and added a note in the section.
Output Signal States During Reset	Updated the second paragraph.

### F.3.4 Reset Clocking and Initialization Revision History

Reference	Description
Power-on reset sequence	Added first bullet.
Single oscillator source clock select	Updated value of cfg_eng_use1 in table.
Single oscillator source reference clock mode	Updated the note.
Clocking	Updated the bit setting for 0010 for CLKCCSR[CLKSEL].
RCW field definitions	Updated the field name SPI_EXT[360-362] in the table "RCW Field Descriptions". Updated the bits RCW[264-266]. Removed the note from RCW[509-511].

### F.3.5 Interrupt Assignment Revision History

No substantive changes

### F.3.6 Arm Modules Revision History

Reference	Description
On-Chip RAM memory controller (OCRAM)	Removed the section from the chapter and added it to the chapter <a href="#">..</a>

## F.3.7 CSU, OCRAM, and MSCM Revision History

Reference	Description
New Chapter	Created a new chapter and the content is extracted from chapter "Secure Boot and Trust Architecture 2.1".
CSU	Updated the endianess of CSU registers.

## F.3.8 System Counter Revision History

No substantive changes

## F.3.9 Interconnect Fabric Revision History

No substantive changes

## F.3.10 CCI-400 Revision History

Reference	Description
LS1043A CCI module integration	Updated the register offset start for CCI-400.
Snoop Control Registers (Snoop_Control_Register_S0 - Snoop_Control_Register_S4)	Updated Snoop Controls (Snoop_Control_Register_Sa) for bit 0.

## F.3.11 TrustZone Address Space Controller (TZA-380) Revision History

No substantive changes

## F.3.12 Supplemental Configuration Unit Revision History

Reference	Description
ALTCBAR Register (SCFG_ALTCBAR)	Updated the bit width of ALTCBAR.

### F.3.13 Device Configuration and Pin Control Revision History

Reference	Description
POR Status Register 1 (DCFG_CCSR_PORSR1)	Removed internal information in bits ENG0, ENG1, and ENG2.
IFC Clock Disable Register (DCFG_CCSR_IFCCLKDR), and eSDHC Polarity Configuration Register (DCFG_CCSR_SDHPCR)	Added the registers.
System Version Register (DCFG_CCSR_SVR)	Updated field [VAR_PER].
Reset Request Mask Register (DCFG_CCSR_RSTRQMR1)	Updated the bit setting for CORE_WDOG3_RST_MSK, CORE_WDOG3_RST_MSK, CORE_WDOG3_RST_MSK, and ALTCBAR_MSK bits.
Reset Request Status Register (DCFG_CCSR_RSTRQSR1)	Updated the bit setting for ALTCBAR_RR bit.

### F.3.14 RCPM Revision History

No substantive changes

### F.3.15 QUICC Engine Block Revision History

No substantive changes

### F.3.16 Data Path Acceleration Architecture (DPAA) Overview Revision History

No substantive changes

### F.3.17 Secure Boot and Trust Architecture Revision History

Reference	Description
CSU and MSCM	Removed the sections from the chapter and added to <a href="#">CSU, OCRAM, and MSCM</a>
Trust architecture objectives	Removed the existing content and added content on the overview of the chapter.

## F.3.18 DDR Revision History

Reference	Description
throughout the chapter	Added data bus width support to 16-/20.
<a href="#">DDR Features</a>	Removed "Partial array self refresh support".
<a href="#">Single-Bit ECC memory error management (ERR_SBE)</a>	Removed "This counter is only incremented when single-bit errors which are not automatically fixed by the controller are detected" from description of SBEC.
<a href="#">DDR SDRAM Address Multiplexing</a>	Added example tables for DDR4.
<a href="#">DDR SDRAM control configuration (DDR_SDRA_M_CFG)</a>	Removed "Note that RD_EN and T2_EN must not both be set at the same time" from DDR_SDRAM_CFG[T2_EN] bit description.
<a href="#">Initialization/Application Information</a>	Removed RD_EN from DDR_SDRAM_CFG.
<a href="#">DDR SDRAM control configuration 2 (DDR_SDRAM_CFG_2)</a>	Changed bit 3 to reserved.
<a href="#">Error Checking and Correcting (ECC)</a>	Reversed the order of Syndrome bits to lsb0 in both tables.
<a href="#">Programming Summary</a>	Updated description of WRLVL_EN to "Should be set to 1 as write leveling is recommended. All other fields in DDR_WRLVL_CNTL, DDR_WRLVL_CNTL_2, and DDR_WRLVL_CNTL_3 should be programmed appropriately based on the DRAM specifications and board layout specifics."
<a href="#">Error Checking and Correcting (ECC)</a>	Updated description of WRLVL_EN to "Should be set to 1 as write leveling is recommended. All other fields in DDR_WRLVL_CNTL, DDR_WRLVL_CNTL_2, and DDR_WRLVL_CNTL_3 should be programmed appropriately based on the DRAM specifications and board layout specifics."
<a href="#">Initialization/Application Information</a>	Removed PASR_DEF and PASR_CFG.
<a href="#">DDR SDRAM Address Multiplexing</a>	Reversed the order of bits to lsb0 in all tables.
<a href="#">DDR register descriptions</a>	Updated description of ESDMODE and SDMODE bits for little endian format.
<a href="#">DDR register descriptions</a>	Removed CS0_CONFIG_2 - CS3_CONFIG_2 as DDR Partial Array Self refresh is not supported.
<a href="#">Initialization/Application Information</a>	Removed PASR_DEF and PASR_CFG.
<a href="#">DDR SDRAM timing configuration 2 (TIMING_CFG_2)</a>	Changed bits 9-12 to Reserved.
<a href="#">DDR register descriptions</a>	In the description of CS1_CONFIG - CS3_CONFIG, BA_BITS_CS changed 01b to Reserved.

## F.3.19 Direct Memory Access Multiplexer (DMAMUX) Revision History

No substantive changes

## F.3.20 DUART Revision History

No substantive changes

## F.3.21 Enhanced Direct Memory Access (eDMA) Revision History

No substantive changes

## F.3.22 eSDHC Revision History

Reference	Description
eSDHC features summary	Removed the bullet saying "Operates only at 1.8 V".

## F.3.23 FlexTimer Module (FTM) Revision History

No substantive changes

## F.3.24 GPIO Revision History

No substantive changes

## F.3.25 Integrated Flash Controller (IFC) Revision History

Reference	Description
Normal GPCM external termination-based program operation	Updated row for IFC_WE0_B.
External signal descriptions	Updated the third paragraph.
Figure 25-64	Updated the figure.

## F.3.26 Inter-Integrated Circuit (I2C) Revision History

No substantive changes

### F.3.27 Low Power Universal asynchronous receiver/ transmitter (LPUART) Revision History

No substantive changes

### F.3.28 PCI Express Interface Controller Revision History

Reference	Description
Features	updated the list to support "x4, x2, and x1 link support."
Lane Reversal	Updated the section per x4 controller lane reversal.
Lane Reversal	Added description of x2 lane reversal support.
PCI Express Device Control 2 Register (Device_Control_2_Register)	Added IDO_REQ_EN (bit 8) and IDO_CPL_EN (bit 9) fields.

### F.3.29 Pre-Boot Loader (PBL) Revision History

No substantive changes

### F.3.30 Quad Serial Peripheral Interface (QuadSPI) Revision History

No substantive changes

### F.3.31 QDMA Revision History

Reference	Description
qDMA Command Descriptor Formats through The qDMA Scatter Gather Table Format	Updated endianness of all the descriptors.

### F.3.32 SATA Revision History

No substantive changes

### F.3.33 SerDes Revision History

Reference	Description
SerDes protocols	Added a new protocol 2560 and 3560. Updated 2233 for sg.m2.
Frame manager MACs	Updated the number of MACs to seven.
SerDes register descriptions	Updated bit PEXA_CFG[1-3] and PEXC_CFG[9-11] for register "Protocol Configuration Register 0 (PCCR0)."
SerDes register descriptions	Updated bit setting 001b for SGMIIB_CFG[5-7] and SGMIIC_CFG[9-11] bits in register "Protocol Configuration Register 8 (PCCR8)."

### F.3.34 Serial Peripheral Interface (SPI) Revision History

No substantive changes

### F.3.35 Thermal Monitor Unit Revision History

Reference	Description
IP block revision register 0 (IPBRR0)	Removed IP block revision register 0 (IPBRR0).
Initialization Information	Updated the fifth step on programming the calibration table. Updated the temperature calibration points table.
TMU temperature range 0 control register (TTR0CR), TMU temperature range 1 control register (TTR1CR), TMU temperature range 2 control register (TTR2CR), and TMU temperature range 3 control register (TTR3CR)	Updated the table "Default Temperature Configuration Points". Updated the reset value of register.

### F.3.36 USB Revision History

Reference	Description
USB PHY Memory Map/Register Definition	Removed registers and bitfields except SUP_IDCODE_HI, SUP_IDCODE_LO, MPLL_LOOP_CTL bits 4-7 and LANE0RX_OVERD_IN_HI bits 6-11.
Global Transmit FIFO Size Register (USB_GTXFIFOSIZn) and Global Receive FIFO Size Register (USB_GRXFIFOSIZn)	Updated the offset addresses.

### F.3.37 WDOG Revision History

No substantive changes

## F.4 Substantive changes from revision 2 to revision 3

Substantive changes from revision 2 to revision 3 are as follows:

### F.4.1 Memory Map Revision History

Reference	Description
CCSR address map	Updated endianness of SEC and security monitor modules.

### F.4.2 Signal Descriptions Revision History

Reference	Description
SPI, eSDHC, USB and GPIO2 signal multiplexing	Updated the RCW[SPI_EXT] setting for RCW[SPI_BASE]=00. Added the following note: "8 bit MMC DDR is not supported."
Signals overview	Added the following note in the JTAG_BSR_VSEL signal description: When this pin is tied high, the USB PHY will not come out or reset in normal functional mode.

### F.4.3 Reset, Clocking, and Initialization Revision History

Reference	Description
RCW field definitions	Updated the description of RCW[SPI_BASE] and RCW[SPI_EXT] bits. Updated the SYSCLK_FREQ bit description to remove support of 125 MHz and 133.33 MHz frequencies. Updated RCW[296-298] as reserved.
RCW field definitions	Added the following note in the RCW[IFC_GRP_A_EXT] and RCW[IFC_GRP_F_EXT] bits: If QuadSPI is selected for cfg_rcw_src then this bit must be selected to 001.

*Table continues on the next page...*

#### Substantive changes from revision 2 to revision 3

Reference	Description
RCW settings for hard-coded Options	Added settings for 0x9F hard-coded RCW option.
External signal descriptions and External signal descriptions	Removed last columns of the tables.

### F.4.4 Interrupt Assignments Revision History

Reference	Description
Internal interrupt sources	Updated interrupt assignments for 143-145, 148, 153-155, 158, 187-189, and 192.

### F.4.5 Arm Modules Revision History

Reference	Description
Arm® Cortex®-A53 core	Added the core cluster sub-system implementation details.
Arm generic interrupt controller (GIC-400)	Added a new table on "GIC memory allocation".

### F.4.6 SCFG Revision History

Reference	Description
SCFG Memory Map/Register Definition	Added the following registers: GIC400_ADDR_ALIGN_64K, G0MSIR1- G0MSIR4, G1MSIR1-G1MSIR4, and G2MSIR1-G2MSIR4.

### F.4.7 Device Configuration and Pin Control Revision History

Reference	Description
System Version Register (DCFG_CCSR_SVR)	Updated SVR[MINOR_REV] and SVR[VAR_PER] bits.

## F.4.8 DDR Revision History

**Table F-1. DDR Revision History**

Reference	Description
<a href="#">DDR SDRAM timing configuration 3 (TIMING_C_FG_3)</a>	In the EXT_REFREC bit description, updated setting 0b101110 to show value of 736 clocks.

## F.4.9 DUART Revision History

Reference	Description
<a href="#">DUART register descriptions</a>	Updated the bit position of all registers.
<a href="#">The DUART module as implemented on the chip</a>	Added this section.

## F.4.10 eSDHC Revision History

Reference	Description
<a href="#">Suspend</a>	Reordered Suspend sequence steps to move step 5 (Save the context registers.....) next to step 1. Moved "After the BGE bit is set" from step 3 to step 2.
<a href="#">eSDHC features summary</a>	Removed bullet for card bus clock frequency.
<a href="#">Data transfer modes</a>	Removed frequencies for all supported modes. Added the following note: "8-bit MMC DDR mode is not supported."
<a href="#">eSDHC register descriptions</a>	Updated the bit positions of all registers.
Throughout	Updated the SDHC data line from 4 to 8.

## F.4.11 Integrated Flash Controller (IFC) Revision History

Reference	Description
<a href="#">Transceiver enable during boot</a>	Updated the second bullet.

## F.4.12 PCI Express Revision History

Reference	Description
PCI Express MSI implementation	Added a figure on MSI implementation.
PCI Express Revision ID Register (Revision_ID_Register)	Updated the register description to add revision values for Si 1.1 and Si 1.0.

## F.4.13 QuadSPI Revision History

Reference	Description
LS1043A QuadSPI module special consideration	Added flexible buffer size value (1 KB).
Throughout	Removed instances of Multicast mode.
Throughout	Removed instances of Multicast mode.

## F.4.14 Universal Serial Bus 3.0 Interface Revision History

Reference	Description
Global Core Control Register (USB_GCTL)	Added the bit CORESOFTRESET.
Power management overview	Removed the second paragraph.

## F.4.15 WDOG Revision History

Reference	Description
Figure 37-4	Updated this diagram.
Watchdog Control Register (WDOG_WCR)	Updated the WCR register description and added the WCR[WDZST] bit.

## F.4.16 Appendix LS1043A (23x23 package) Revision History

Reference	Description
Register differences	Updated the SVR and Device ID description for 23x23 package.
Signals Overview	Added the following note in the JTAG_BSR_VSEL signal description: When this pin is tied high, the USB PHY will not come out or reset in normal functional mode.

## F.5 Substantive changes from revision 1 to revision 2

Substantive changes from revision 1 to revision 2 are as follows:

### F.5.1 Memory Map Revision History

Reference	Description
DDR remapping	Added a note.
Memory map overview	Added a list item "Internal debug control and status ....."

### F.5.2 Signals Description Revision History

Reference	Description
Signals overview	Removed the ccp signals as well as esdxc_activity_gate.
Configuration signals sampled at reset	Added cfg_eng_use1 and cfg_eng_use2.
Signals overview	Updated the description of JTAG_BSR_VSEL and TBSCAN_EN_B signals.

### F.5.3 Reset, Clocking, and Initialization Revision History

Reference	Description
Figure 4-4	Updated the figure.
Single source clocking	Updated the section.
Single oscillator source reference clock mode	Updated the first paragraph.
DRAM type select	Updated the "DRAM type" for DDR4.
Reset configuration word (RCW)	Added the note.
Figure 4-3, Figure 4-5, and Figure 4-4	Updated the figure by adding FMAN-MAC.
Single oscillator source reference clock mode	Added the note.
IP logic clock distribution and configuration	Added the note. Updated the figures.
RCW field definitions	Updated the description of RCW[509-511].

## F.5.4 Interrupt Assignments Revision History

Reference	Description
Internal interrupt sources	Updated interrupt assignments for 143-145, 148, 153-155, 158, 187-189, and 192.
Internal interrupt sources	Added clarity to the existing content on interrupt 196, 197, 200, and 201.

## F.5.5 Arm Modules Revision History

Reference	Description
Arm generic interrupt controller (GIC-400)	Added a new table on "GIC memory allocation".
On-Chip RAM memory controller (OCRAM)	Updated the note.

## F.5.6 System Counter Revision History

Reference	Description
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.7 Interconnect Fabric Revision History

Reference	Description
IF	Removed registers from 2CA_0008 to 2CA_2618 and 2880 to 288C. Removed instances of SMMU.

## F.5.8 Cache Coherent Interconnect (CCI-400) Revision History

Reference	Description
Security	Consolidated the list items.
QoS Regulator Scale Factor Registers ( <a href="#">Latency_Regulation_Register_S0</a> - <a href="#">Latency_Regulation_Register_S4</a> )	Represented the scale factor in terms of power of 2 in register "QoS Regulator Scale Factors".

## F.5.9 Arm CoreLink™ TrustZone Address Space Controller TZC-380 Revision History

Reference	Description
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.10 SCFG Revision History

Reference	Description
SCFG	Added new registers "GIC400_ADDR_ALIGN_64K". Added new registers: G0MSIR1- G0MSIR4, G1MSIR1- G1MSIR4, and G2MSIR1- G2MSIR4.
Extended RCW PinMux Control Register (SCFG_RCWPMUXCR0)	For bits IIC3_SCL, IIC3_SDA, and IIC4_SCL, replaced bit setting for 111 with "Reserved".

## F.5.11 Device Configuration and Pin Control Revision History

Reference	Description
System Version Register (DCFG_CCSR_SVR)	Updated the SVR value.

## F.5.12 Run Control and Power Management (RCPM) Revision History

Reference	Description
Power Management Control and Status Register (RCPM_POWMGTCSR)	Removed the note "If debugger wants.....".
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.13 QUICC Engine Overview Revision History

Reference	Description
Figure 15-2	Updated the data paths block diagram for the correct number of cores.

## F.5.14 DDR Revision History

Reference	Description
DDR SDRAM control configuration 3 (DDR_SDRAM_CFG_3)	Added note to descriptions of bit fields ECC_FIX_EN and ECC_SCRUB_INT 'Scrubbing cannot be enabled until after the controller has cleared DDR_SDRAM_CFG_2[D_INIT].'
DDR SDRAM timing configuration 2 (TIMING_CFG_2)	Added note to description of bit field FOUR_ACT.
DDR SDRAM timing configuration 4 (TIMING_CFG_4)	Changed setting 0b10 of bit field DLL_LOCK from Reserved to 1024 clocks.
Memory Interface Signals	Removed mention of [unsupported] CRC in description of MAPAR_ERR_B.

## F.5.15 DUART Revision History

Reference	Description
DUART register descriptions	Corrected the offsets of UART2.

## F.5.16 eDMA Revision History

Reference	Description
Error Status Register (DMA_ES)	In description, replaced "See the Error Reporting and Handling section" with "See Fault reporting and handling for more details."
Enable Asynchronous Request in Stop Register (DMA_EARS)	Removed the register.
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.17 eSDHC Revision History

Reference	Description
System Control Register when ESDHCCTL[CRS=0] (SYSC TL_ESDHCTL_CRS_0)	Updated the reset value of bit DVS to 0x3 in "System control register when ESDHCCTL[CRS=0] (SYSC TL_ESDHCTL_CRS_0)" register.
eSDHC register descriptions	Updated the bit ordering of all the registers from little endian to big endian.
Protocol control register (PROCTL)	Made bit PROCTL[0] reserved.
Overview	Updated the section.

## F.5.18 Integrated Flash Controller (IFC) Revision History

Reference	Description
Internal connectivity of RB signal	In second para added "Note that ,If IFC_CS_B[5] is used, then IFC_RB_B[3] cannot be used and vice-versa."
Transceiver enable during boot	Updated the second bullet.

## F.5.19 PCI Express Revision History

Reference	Description
PCI Express MSI implementation	Added a figure on MSI implementation.
PCI Express Revision ID Register (Revision_ID_Register)	Updated the reset value to 01.
PEX_LUT register descriptions	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.20 Pre-Boot Loader Revision History

Reference	Description
Features summary	Removed all the references to voltage for eSDHC SD/eMMC interface.

## F.5.21 QuadSPI Revision History

Reference	Description
<a href="#">QuadSPI register reset values</a>	Added a new topic stating the chip specific reset values of the registers.
<a href="#">Driving External Signals</a>	Updated the figure "Serial Flash Access Scheme". Updated the characteristic "IDLE" of QuadSPI module.
<a href="#">Byte Ordering - Endianness</a>	Updated the section from 32 bit LE format to 32 bit BE format.
<a href="#">Peripheral Bus Register Descriptions</a>	Updated the registers to big endian format.
<a href="#">Flash Programming</a>	Updated TX Buffer up to a maximum of 16 from 32.
<a href="#">TX Buffer Status Register (QuadSPI_TBSR)</a>	Updated the register bits and their width from 16 to 24.
<a href="#">TX Buffer Data Register (QuadSPI_TBDR)</a>	Updated the first TX buffer depth from 128 to 64 bytes.
<a href="#">AMBA Bus Register Memory Map</a>	Updated SFADR[23:0] to SFADR[8:31] and SFADR[31:0] to SFADR[0:31] in the fifth paragraph of note. Updated SFADR[24:1] to SFADR[7:30] and SFADR[31:1] to SFADR[0:30] in the last paragraph of note.

## F.5.22 Queue Direct Memory Access Controller Revision History

Reference	Description
Throughout the chapter	Updated the bit ordering of registers to represent in decrementing order (31-0).

## F.5.23 SATA3.0 Revision History

Reference	Description
<a href="#">PhyControl BIST modes</a>	Added "Seven" in beginning of first step.
<a href="#">Port config register (PCFG)</a>	In bit description of [TPSS] and [TPRS], updated 'millisecond' as 'microsecond'.
<a href="#">Port PhyControl status register (PPCS)</a>	Updated register description under 'Function' section.
<a href="#">Port Phy2Cfg register (PP2C) and Port Phy5Cfg register (PP5C)</a>	Updated the register descriptions to remove OOB specification calculation example.
<a href="#">HBA capabilities extended register (CAP2), Timer control register (TCR), Port x command and status register (PxCMD), Port x SATA status register (PxSSTS), and Port x SATA control register (PxSCTL)</a>	Updated the following registers to remove the DevSleep feature support: Updated CAP2[5-3] bits as reserved. Updated the TCR register description to remove reference of DevSleep timer. Updated PxCMD[ICC] bit setting 1000b to reserved. Updated PxSSTS[IPM] bit setting 1000b to reserved.

Table continues on the next page...

Reference	Description
	Updated PxSCTL[IPM] bit settings 0100b-0111b to reserved.
AXI PROT control register	Removed the AXI PROT control register (AXIPC) from offset C4h.
HBA capabilities extended register (CAP2), Timer control register (TCR), Port PhyControl status register (PPCS), Port x SATA error register (PxSERR)	Updated the reset value of CAP2 register to 0x0000_000C. Updated the reset value of TCR register to 0x0000_0100. Updated the reset values of PPCS and PxSERR registers to undefined.
Modifications to the AHCI standard PRDT entry	Updated this section to remove references to AXI PROT control register.

## F.5.24 Secure Boot and Trust Architecture Revision History

Reference	Description
TrustZone Watchdog (TrustZone WDOG)	Updated last paragraph to remove statement specific to deep sleep modes.
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.25 SerDes Module Revision History

Reference	Description
Valid reference clocks and PLL configurations for SerDes protocols	Added a second note below the table "Valid SerDes RCW Encodings and Reference Clocks".
SerDes register descriptions	Revised the field description of SGMIIaCR1[MDEV_PORT].
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31).

## F.5.26 Universal Serial Bus 3.0 Interface Revision History

Reference	Description
Global User Control Register 1 (USB_GUCTL1)	Updated the bit setting of USBx_GUCTL1[L1_SUSP_THRLD_FOR_HOST].
Global Core Control Register (USB_GCTL)	Updated the bit setting of USBx_GCTL[PRTCAPDIR].

Substantive changes from revision 0 to revision 1

## F.5.27 WDOG Revision History

Reference	Description
Flow Diagrams	Updated the figure "Time-Out Counter Flow Diagram".
Throughout the chapter	Updated the bit ordering of registers to represent in incrementing order (0-31). "

## F.5.28 Appendix LS1043A (23x23 package) Revision History

Reference	Description
Signals Overview	Updated the signal description of TBSCAN_EN and JTAG_BSR_VSEL.

## F.6 Substantive changes from revision 0 to revision 1

Substantive changes from revision 0 to revision 1 are as follows:

### F.6.1 Overview Revision History

Reference	Description
Quad Serial Peripheral Interface (QuadSPI)	Removed "Maximum serial clock frequency 62.5 MHz."
Features summary	Added the support of full-duplex mode in SerDes lanes for high-speed peripheral interfaces.
Serial ATA (SATA) controller	Added a note pertaining to hot-plug capabilities.

### F.6.2 Signal Descriptions Revision History

Reference	Description
SPI, eSDHC, USB and GPIO2 signal multiplexing	Updated the SPI_CLK signal configuration.

## F.6.3 Reset, Clocking, and Initialization Revision History

Reference	Description
RCW field definitions	Updated HOST_ AGT_PEX, SPI_EXT, and SerDes_REFCLK_SEL.

## F.6.4 Interrupt Assignments Revision History

Reference	Description
Internal interrupt sources	Added clarity to the existing content on interrupt 140.

## F.6.5 SCFG Revision History

Reference	Description
, , , , and	Updated register names and descriptions to bring clarity that these registers are not tied to a specific PCI express controller, but can be used by any PCI express controller.

## F.6.6 Device Configuration and Pin Control Revision History

Reference	Description
Fuse Status Register (DCFG_CCSR_FUSESR)	Removed fuse status register.
DCSR register definition	Removed the registers of DCSR space including: <ul style="list-style-type: none"> <li>• Single bit ECC status register 1 (DCSR_DCFG_SBEESR1)</li> <li>• Single bit ECC status register 2 (DCSR_DCFG_SBEESR2)</li> <li>• Multi-bit ECC status register 1 (DCSR_DCFG_MBEESR1)</li> <li>• Multi-bit ECC status register 1 (DCSR_DCFG_MBEESR2)</li> </ul>

## F.6.7 Run Control and Power Management (RCPM) Revision History

Reference	Description
Throughout	Overhauled this chapter which includes: <ul style="list-style-type: none"> <li>• Update to overview, Power Management State Summary, Modes Entry and Exit for Power Management</li> </ul>

*Table continues on the next page...*

## Substantive changes from revision 0 to revision 1

Reference	Description
	<ul style="list-style-type: none"> <li>Merged section "Device LPM20 Operation" and "LPM20 State" and overview of sections "Core power management" and 'Device power management'.</li> <li>Removed "Initialization Information" and "Application Information"</li> <li>Replaced STANDBYWFI(PW15) to PH20 in section "LPM20 State".</li> </ul>
Power Management State Summary	<p>Removed the column "status visible" from table "Core and Cluster Power Management State Summary" and "Device Power Management State Summary".</p> <p>Updated core clock to "Y" for full on power mode.</p>
Through out	Added new power mode SWLPM20.

## F.6.8 QUICC Engine Overview Revision History

Reference	Description
Table 15-6	Updated the EXTC bit description.

## F.6.9 DDR Revision History

Reference	Description
DDR Signals Overview	Replaced reset value of MDIC with reference to datasheet
DDR SDRAM timing configuration 3 (TIMING_C FG_3)	In bitfield CNTL_ADJ, updated all non-reserved settings except 000.
Through out the chapter	<p>Added support to 16-/20-bit data bus width which includes:</p> <ul style="list-style-type: none"> <li>Addition of 16-/20-bit in data bus width supported.</li> <li>Update to bit setting for DDR_SDRAM_CFG[DBW].</li> <li>Update to description og DDR_SDRAM_CFG[T3_EN].</li> <li>Update to CAPTURE_ECC[ECE].</li> </ul>

## F.6.10 DUART Revision History

Reference	Description
DUART register descriptions	Removed the duplicate rows from platform frequency 333 MHZ in table "Baud Rate Examples".

## F.6.11 Integrated Flash Controller (IFC) Revision History

Reference	Description
Normal GPCM external termination-based program operation	Added sentence saying "IFC is not available for further transactions until IFC_TA is deasserted".
Figure 25-50	Updated second last signal from "read_data_mem" to 'AD'.
Normal GPCM external termination-based read operation	Removed "Normal GPCM access termination with IFCTA (write access)" figure.
External signal descriptions	Provided clarity to AD[0:n] by adding explanation towards end of description.
NAND asynchronous mode program data timing	Updated the figure to reflect CLE and ALE high time as TWP+TWCHT.

## F.6.12 qDMA Revision History

Reference	Description
Memory Map	Updated from "...virtualized 4KB pages" to "...virtualized 64KB pages". Updated the address offset in table "qDMA Command/Status Queue Mapping in CCSR Space".
qDMA register descriptions	Added a note for register "DMA status register (DSR_M)."

## F.6.13 SATA Revision History

Reference	Description
SATA features summary	Updated the last bullet as SATA Vendor BIST mode.

## F.6.14 Secure Boot and Trust Architecture Revision History

Reference	Description
Terminology used in this	Under the Non-Secure World bullet, updated "NS=0" to "NS=1". Under Non-Secure state bullet, updated RCW[SB_EN] value to 0.
Config Security Level (CSU CSL)	Added a note.

## F.6.15 SerDes Revision History

Reference	Description
Disabling unused SerDes modules	updated the configuration to "SRDS_PRTCL_S1 = 0x0000".
The SerDes module as implemented on the chip	Added note to state that only full-duplex mode is supported on all Ethernet interfaces.
MDIO register spaces	Restructured the complete section on MDIO Register Descriptions and removed the support to half duplex mode.
Frame manager MACs	Removed the support of MAC10 from table "MAC Capabilities".

## F.6.16 Thermal Management Unit Revision History

Reference	Description
IP Block Revision register 1 (TMU_IPBRR1)	Removed IP Block Revision register 1 (TMU_IPBRR1).
TMU mode register (TMR)	Updated TMR[MSITE] bit settings.
The TMU module as implemented on the chip	Added a note stating "The maximum operating temperature...".
TMU mode register (TMR)	Updated TMR[ALPF] bit description.
TMU status register (TSR)	Updated the register description.
TMU sensor configuration register (TSCFGR)	
TMU temperature configuration register (TTCFGR)	
TMU temperature range 0 control register (TTR0CR)	Removed the note and updated the table to its default value.
TMU temperature range 1 control register (TTR1CR)	
TMU temperature range 2 control register (TTR2CR)	
TMU temperature range 3 control register (TTR3CR)	
Initialization Information	Updated TTCFGR value for TTR0CR, 12 points at 0°C. Added point 0, 1, 2.... headers in the table.
TMU monitor temperature measurement interval register (TMTMIR)	Updated the paragraph below note "The temperature measurement interval....." and the table "Temperature measurement interval".
Modes of Operation	Added the table.

## F.6.17 Appendix LS1043A (23x23 package) Revision History

Reference	Description
<a href="#">Introduction</a>	Added the appendix.



**How to Reach Us:**

**Home Page:**

[nxp.com](http://nxp.com)

**Web Support:**

[nxp.com/support](http://nxp.com/support)

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [nxp.com/SalesTermsandConditions](http://nxp.com/SalesTermsandConditions).

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS,ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamiQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2016–2020 NXP B.V.

Document Number LS1043ARM  
Revision 6, 07/2020

